1984

A pseudo colour image display system.

Pradeep Kumar. Goyal
University of Windsor

CANADIAN THESES

THÈSES CANADIENNES

NOTICE

The quality of this microfiche is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this film is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30. Please read the authorization forms which accompany this thesis.

AVIS

La qualité de cette microfiche dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, examens publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de ce microfilm est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30. Veuillez prendre connaissance des formules d'autorisation qui accompagnent cette thèse.

THIS DISSERTATION
HAS BEEN MICROFILMED
EXACTLY AS RECEIVED

LA THÈSE A ÉTÉ
MICROFILMÉE TELLE QUE
NOUS L'AVONS REÇUE

Canada

A Pseudo Colour Image Display System.

by

Pradeep Kumar Goyal

A thesis
presented to the University of Windsor
in partial fulfillment of the
requirements for the degree of
Master of Applied Science
in
The Department of Electrical Engineering
University of Windsor

Windsor , Ontario, 1984

ABSTRACT

The goal of this project is to develop a 'user friendly' system for display purposes in a Digital image processing environment. A scheme to allow access to the computer data via a touch screen (manufactured by TSD Display products, Inc.) mounted on the Aydin 5216 colour display monitor has been developed and implemented. In this scheme 'Soft switches' are generated on the screen under program control. The program accepts the touch position and performs the operation specified by that soft switch. These switches allow the user to display images on the screen or perform certain modifications on the displayed images or graphic data.

This thesis also covers and explains the Aydin 5216 colour display monitor and a software package provided by Aydin to perform editing on the screen. This package has been modified to suit the host computer (SEL 32/27). All the relevent details of the changes made in the package have been explained.

A software scheme for interactive quality control has also been developed as a part of this work. This program processes the image specified by the user to isolate its boundary and thresholded image. The original image and its histogram is also displayed. An additional option has been incorporated in the program through which the threshold level can be chosen either by the user or by the program. If the former option is chosen the threshold level is specified by touching the histogram at the point where the threshold is desired. No interaction with the keyboard is needed at any stage.

This thesis additionally explains a technique to convert monochrome images to colour images. Two different interpretations, namely, 'Density slicing and Grey level to colour transformation, have been presented.

A scheme to produce colours that closely match the original colours of the object ('Real colours') has been investigated. The scheme has not been found to be very effective on this monitor. An analysis for the failure of the scheme has been included in the discussion.

ACKNOWLEDGEMENTS

This is to thank my Supervisors Dr. G.A. Jullien and Dr. W.C. Miller for their guidance. Thanks are also due to Dr. Maher A. Sid-Ahmed for his valuable suggestions, help and continued encouragement.

I also wish to thank Mr. M.V. Prasada Rao, Mr. A.S. Sethi and all other graduate students for their moral support.

My humble respects go to my Parents for their continued faith and confidence in me.

And above all, my sincerest gratitude and regards go to Dr. M. Shridhar and Dr. M. Ahmadi for their support and encouragement during a very difficult period.

CONTENTS

LIST OF FIGURES

LIST OF PHOTOGRAPHS

Chapter I

THE SYSTEM DESCRIPTION

1.1 INTRODUCTION

An image processing station has been designed and developed in this department capable of processing images in pseudo colour. The work has been accomplished as a group effort. This thesis deals with the display system and the various aspects associated with it. Software has been developed as a part of this work for a touch switch control from the monitor. A scheme for interactive quality control has also been proposed and implemented.

The goals and objectives of this work are defined as follows:

1. Implementing a Fortran Software package on the system which is capable of performing modifications on the displayed image. (like Zoom, Lookup Table generation).

2. To develop a scheme to use this package.

3. To generate a Soft switch feedback system on the screen. This involves making a SEL Executive for editing images on the screen in order to make the system user friendly.

4. Demonstrating the effectiveness of the system for an Interactive quality control process.

5. Development of software for displaying images in Pseudo Colour and Real colour on the Aydin monitor.

1.2 THESIS ORGANISATION

The total System explained is made up of the host Computer,[1] Colour Display Terminal,[2] Colour Printer,[3] Camera,[4] Convolver[5] and a Touch Screen[6] mounted on the display monitor. A brief introduction of the system has been provided followed by the description of the Firmware of the Aydin colour monitor. Various techniques for transformation of

---

[1] SEL Computer is manufactured by Gould, Inc.

[2] Colour terminal is manufactured by Aydin Controls, Inc.

[3] Colour Printer is manufactured by Triloq Corporation, Inc.

[4] The Camera is manufactured by Hamamatsu Co. of Japan. It is a special purpose camera designed for Scientific Applications.

[5] This convolver has been designed in the department of Electrical Engineering, University of Windsor.

[6] Touch Screen is manufactured by TSD products, Inc.

Grey level images to colour images have been explained and implemented [1,9]. This has been provided because all the storage and the image pictures are processed in grey level and only at the display stage these monochrome images are converted into colour images. The communication scheme between the display monitor and the computer is also presented.

Various software techniques are developed for the system. These include : the graphics software package, SEI Executive structure, Colour matching between printer and Aydin, histogram plotting, selective zooming, Interactive Quality Control monitoring and investigation into Real colour production on the Aydin monitor.

1.3 IMAGE PROCESSING SYSTEM

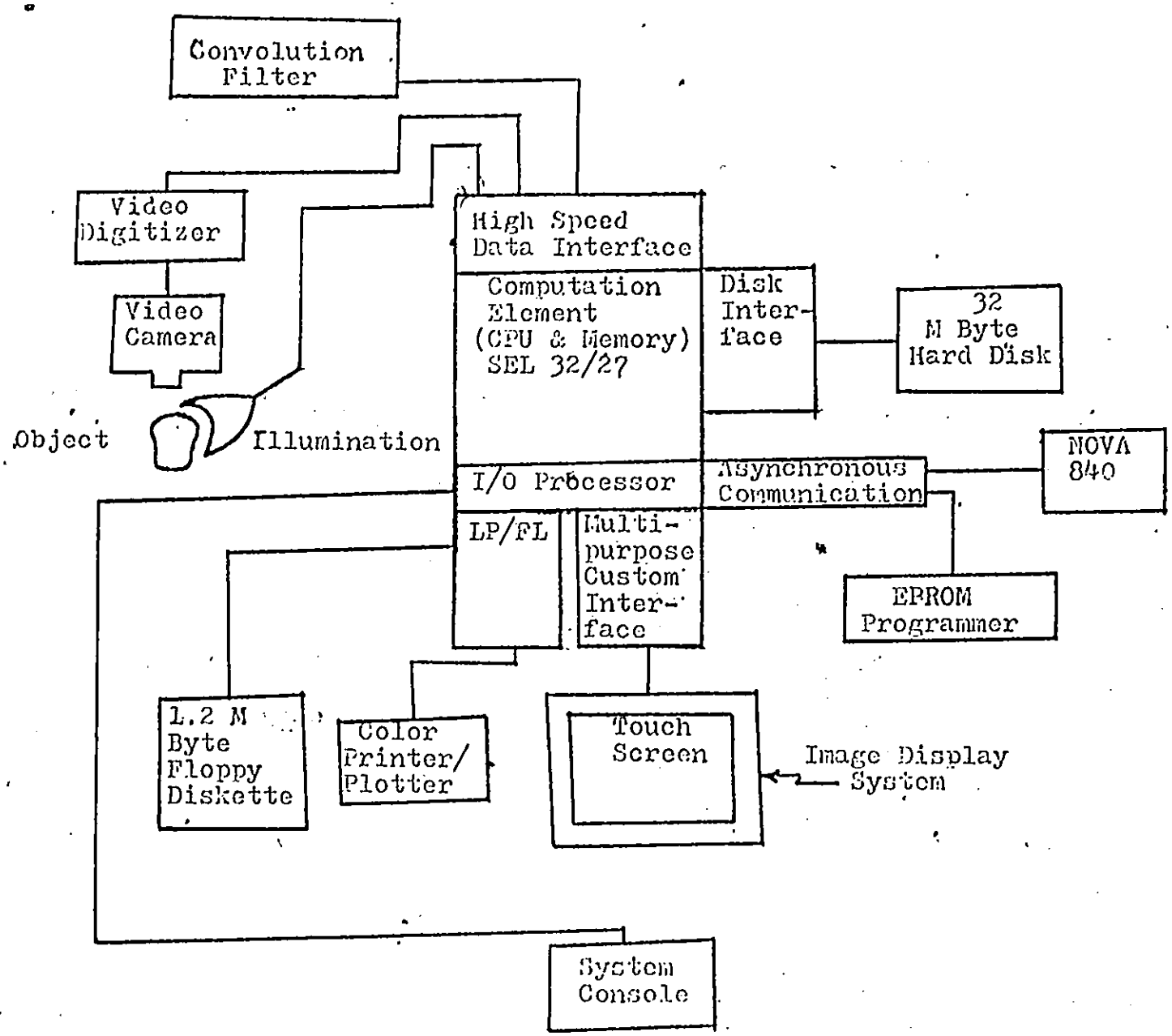
It has been estimated that over half the information that we perceive through our senses is by way of vision [13]. It is thus obvious that the most natural form of communication between man and machine is by way of visual information transfer. This field had attracted little attention in the past because of several reasons which include the large amount of computation and memory that is normally required for picture processing. But in the last decade, memory and the hardware costs have decreased and computing speeds have increased. Due to these relatively new advances,

Digital Image Processing is a field that is gaining a lot of importance lately.

This Image Processing station with a touch screen mounted on the AYDIN colour monitor is aimed to provide a touch switch control for the operator. The Image Processing system developed in the department is centered around the host computer SEL 32/27 manufactured by the Systems Engineering Laboratories. The image is input to the computer through a video camera and digitizer connected to the Main Bus (SEL BUS) through the High Speed Data Interface (HSD) [14]. Preprocessing of the image is possible by means of a convolution filter Filter [8] which is also interfaced to the SEL BUS through the HSD interface.

Fig [1.1] shows the various units and peripherals linked to the computer.

The processed image can be displayed on the AYDIN Colour Monitor [2,3] which is interfaced to the I/O processor [5] through the Multi Purpose Custom Interface (MPCI) [4]. The hardcopy of the processed image can be obtained on the TRILOG Colour printer [11], which is connected to the I/O processor through the Line Printer/ Floppy interface and Multipurpose Bus.



Organization of the Image Processing System.

The operator can control the process from the System Console connected to the I/O processor through RS232C interface and from the Touch Screen [17], mounted on the AYDIN colour monitor. This system is also connected to the NOVA 840 computer through the Asynchronous Communications Bus to facilitate data transfer between the two computers.

1.4 HOST COMPUTER-SEL 32/27

The SEL computer is a 32 Bit Mini computer and has an Input/Output Processor, a High Speed Data Interface. The hard disk pack, HSD and the CPU communicate through the SEL BUS. The I/O processor communicates between the SEL BUS and the Multipurpose Bus (MP BUS). To facilitate the development of processing algorithms a Multi Tasking Executive (MEX-32) and a FORTRAN 77 compiler have also been incorporated into the system.

The SEL BUS is a high speed 32 bit Synchronous Bus capable of transferring data at the rate of 26.67 MBytes per second. The Multipurpose Bus (MP BUS) on the other hand is a medium speed 16 bit Asynchronous Bus capable of transferring data at the rate of 1.5 M Bytes per second.

1.5 AYDIN 5216 DISPLAY

The AYDIN 5216 Display unit is a monitor with its processing ability built around the INTEL 8086 16-bit microprocessor. The standard Firmware provided with the display has an instruction set of alphanumeric and graphic instructions which accepts and processes instructions for display.

The AYDIN 5216 display computer is interfaced to the SEL 32/27 computer. The standard firmware instruction accepts codes from the host computer and generates alpharumeric or graphic data on the display monitor. This monitor has a picture resolution of 512 by 512 pixels. Depending on the display hardware configuration 256 or 1024 colours[7] may be displayed on the colour monitor simultaneously. In addition to computer control through coded instructions, the 5216 interfaces with the user through its key board inputs.

---

[7] The Aydin monitor can be used with 2 lookup table cards. VID 001 can produce as many as 1024 colours and the data width is 10 bits for this card. The second card is VID 004 capable of producing 256 colours on the screen. The latter card has been incorporated in our system.

Chapter II

THE IMAGE DISPLAY SYSTEM

2.1 INTRODUCTION

In this chapter the Aydin command structure has been explained. The commands included here are the ones which are most frequently used by the user and are necessary for the understanding of the work. Some of the instructions that have not been included can be found in [3].

The 16 bit command code for each of the command is given in Appendix A.

2.2 THE SYSTEM

The firmware provided by the Aydin Company has no operating system of its own and in our configuration it is provided by the host SEL (Systems Engineering Laboratory) Computer [3]. Also provided is an Interface card for Communication between SEL Computer and the Aydin Colour display monitor. The display system is a 512 x 512 Pixel squared, 256 colour resolution terminal. The 8 bits of the data are fed as eight inputs to the three colour guns namely Red,

Green and Blue. The lower three bits drive the red gun, the next three drive the green gun and the last two bits are connected to the blue gun. It is possible to allocate different colour values to a given Pixel through a lookup table.

The terminal accepts Commands either from the host computer or from the monitor's own keyboard. Each Command on the monitor's keyboard has a unique 16 bit code. This code can either be provided by the host computer or by the terminal's keyboard. The system can operate both in Words[8] as well as in the Pixel Mode. The Alpha-numeric display is possible in any one or combinations of 256 colours available to us.

The definitions and explanations of various Commands of the Aydin system follow.

2.3 PARAMETER SET

This message to select the parameter set is to be issued to begin any operation for the Aydin terminal. There are 15 different parameter sets that can be used. Each one of them has its own set of Major and Minor Channel Selects and Masks, Mode Control Words, X and Y Cursor Position, Conic and Rectangular Viewpoints, Foreground and Background

---

[8] Word length is 16 bits.

Pixel Values etc.

The last 4 bits indicate which one of the 15 parameter sets has been selected. In this case it has been standardised to 1. The equivalant of this Command in the Fortran package is Subroutine SELVIW.

2.4 RECTANGULAR LIMITS

The rectangular viewpoint can be adjusted on the screen i.e. instead of a normal 512 x 512 Pixel screen, the user can use only a part of it, and all the functions and editing take place within that zone. The rectangular viewpoints are forced by issuing four limit Commands namely LEL, LRR, LRT and LRB denoting limit rectangular view- point left, right, top and bottom respectively. Each is a 16 bit instruction of which the lower 10 bits denote the limit. After the limits are forced the Cursor does not move out of the limits; instead it wraps around the viewpoint limits. However, if a Command to position the Cursor outside the viewpoint is issued, an error condition results.

Another factor to be taken note of, in this case, is that the Copy Command is not affected by the rectangular limits. The Conics too are not limited by the rectangular limits. They have their special Conic limits.

When the unit is powered on, a full screen viewport is in effect.

2.5 MODE CONTROL WORD (MCW)

If bit 0 is high then it is indicated that the input data is graphic data and bits 2,3,4,6 are ignored. (See the bit pattern of the command in appendix A)

Bit 1 selects the Word or the Pixel Mode. In the Word Mode 16 adjacent Pixel locations are accessed at a time. In case of Alpha-numeric data, the Word Mode is more efficient. In the Pixel Mode the data entry is made using Foreground Pixel Register (FPR) and the Background Pixel Register (BPR).

When Bit 2 is high, the data word following the instruction is unpacked with the higher order Byte first and the lower order Byte later. When bit 2 is low it is the other way around.

In the Alpha-numeric Mode (bit zero = low), bits 3 and 4 of MCW are decoded to select one of the four Character Fonts.

Bits 5,6 and 7 determine how data is entered into the refresh memory channels. The basic difference between modes is the manner in which the data is processed before being written into the refresh channels.

For the Alpha-numeric data, bit 8 determines if the cursor is to be advanced by the values specified by the user, or through adjustable Cursor Advance (ACA) instruction.

2.6 CURSOR

Fifteen separate cursor locations are stored in the 5216 standard firmware. However, only three Cursors can be displayed at any given time. Each one of them is recalled by specifying through the Start of Message command. (Since it has been standardised to parameter set number to 1, in most cases Cursor # 1 will be on the screen.) The Cursor's zero position (Home Position) is at the upper left corner of the rectangular viewpoints. It may or may not be displayed.

The Cursor position can be specified as a relative displacement in X and Y directions or it can be an absolute positioning of the same. If this instruction puts the cursor out of the rectangular viewpoints then an error message is sent to the computer.

2.6.1 ADJUSTABLE CURSOR ADVANCE (ACA)

In some cases it is desirable to have some extra spaces between consecutive characters over and above that provided automatically. This instruction is used to spell out the Cursor advancement after each character. After the character is displayed, it is checked to see if the Cursor is still in the viewpoint limits. This instruction may not be executed unless the ACA flag in the Mode Control Word is set.

2.6.2 CURSOR MOVE

This instruction governs the direction of the movement of the Cursor in the X and Y directions.

2.6.3 LOAD INDEX REGISTER (LIX , LIY)

The least significant bits of LIX and LIY specify the contents of X and Y index registers respectively. One can associate different Cursors with different parameter sets by assigning the Cursor address to another Cursor number.

2.7 ALPHA-NUMERIC DATA DISPLAY

The two ways to enter Alpha-numeric data into the colour monitor are through 'Load Alpha-numeric character' or by 'Block Transfer Mode'. The latter is a more efficient mode to transfer a large set of data, whereas the former only loads a single character.

2.8 GRAPHIC DATA DISPLAY

Several commands are available for pixel entry into the monitor namely, Write Pixel Values (WPX), Write Pixel Foreground, Background (WPXF and WPXB) and Load Graphic Elements (LGE). In order to write conics, one has to use Execute Conic instruction (EXC).

2.8.1 VECTORS

The starting point of the desired vector is loaded into cursor registers and the end point is loaded into the index registers. An Execute Vector command is then issued, causing the display generator to compute and write the necessary points to produce a line segment between start and end points. The cursor is positioned at the end point of the vector.

2.8.2 CIRCLE

The center point is loaded into the cursor registers. The radius or the desired circle is loaded into the X index register. Upon receiving an EXC (Circle) instruction, the display generator computes all the necessary points to write the circle with the given radius and center. The cursor is positioned at the center of the circle. Concentric circles can thus, be drawn by changing the X index and issuing the next EXC instruction.

2.9 CONIC LIMITS

A special set of limits called the conic limits govern the input of conic display data. They are specified by the four Conic Limit instructions : Right, Left, Top and Bottom. New conic data will be entered onto the screen only within conic limits.

The ten least significant bits are the binary values of the conic limits. But since the system is limited to 512 pixel squared screen the tenth bit is kept low all the time.

Since conics are drawn only within the conic limits, arcs can be generated by properly setting these limits before executing a conic instruction.

2.10 BLOCK TRANSFER MODE (BXM)

The fastest way to load either Alpha-numeric or graphic data from the host to the colour monitor is by using the Block Transfer Mode (BXM) instruction. After receiving just one BXM instruction, the colour monitor is prepared to receive up to 2047 words, or by using the Extended BXM (XBXM), up to 2**24-1 data words. Cursor advancement is managed automatically.

Since the BXM instruction is same both in the case of Alpha-numeric and Graphic data, the system checks the state of MCW to determine if the data is to be processed as graphic or alpha-numeric.

2.11 DISPLAY EDIT FUNCTIONS (EDT)

The Edit instruction (EDT) provides for Rolling or Scrolling of data both destructively and non-destructively and Clearing the screen. These operations are only performed within the rectangular window limits.

Other Edit functions are Zoom, Fill and Copy.

2.11.1 SCROLL

Bits 1,2,3 and 4 of EDT define Scroll instruction, which causes the displayed data to move up or down one vertical element. The execution time for the scroll instruction depends on the value of the rectangular limits.

2.11.2 ROLL

Bits 3 and 4 define the Roll instruction. They are defined as follows.

BITS		MEANING
4	3	
0	0	No change
0	1	Roll right
1	0	Roll left
1	1	Undefined

A one in bit 3 or 4 causes the image to move right or left by one horizontal pixel. A high in both the bits is a conflicting situation and causes an error signal.

2.11.3 DESTRUCT / NON - DESTRUCT

When bit 5 is set to 0, any data which is scrolled or rolled beyond the limits of the rectangular window wraps around.

When bit 5 is low, the data which is rolled or scrolled beyond the rectangular window is lost.

2.11.4 CLEAR

Bit 1 of EDT specifies that a clear operation is to be performed. The screen is cleared only within the rectangular limits.

2.11.5 ZOOM

Upon execution of this instruction, starting from the center of the window all distances from pixel to center are doubled. Any data pushed outside the window limits is lost.

2.11.6 FILL

This instruction is used to fill an area with a new pixel value. The area is bounded by any pixel which does not equal the pixel value over which the cursor was located when the instruction was executed.

The new pixel value with which the area to be filled is the value in the Foreground Pixel Register (FPR).

2.11.7 COPY

It copies pixel data from one rectangular area of the screen to another. The area to be copied is defined by the conic limits. The area to be copied from has the same size as the conic limits, and is located on the screen using the cursor position as its center.

The copy function is instructed as to which pixels are to be copied by specifying either a particular pixel or a range of pixel values.

2.12 DATA TRANSMISSION TO HOST

The monitors Firmware provides functions to enable data transmission to the host computer. The position of any of the 15 cursor coordinates and the contents of Video look-up table can be requested by the host computer.

Also, keystroke data typed on the keyboard (5115) can be transmitted to the host.

2.12.1 CURSOR POSITION READBACK (TCO)

After this instruction is sent, the cursor position stored in the parameter set specified by the lower 4 bits of TCO is sent to the host.

2.12.2 MEMORY DATA (TSC) AND EXTENDED MEMORY DATA (XTSC) READBACK

These 2 functions cause the 5216 to transmit refresh memory data to the host computer. Data is transmitted starting from the upper left corner, regardless of the cursor position, and proceeds left to right, top to bottom.

It should be noted, however, that the reading of data takes place only within the rectangular limits.

Data is transmitted in BXM graphic word mode form regardless of the MCW status at the time the command was issued.

2.12.3 END OF TRANSMISSION (EOT)

All data transmission from the 5216 to the host computer is automatically formatted by the 5216 to include an End of Transmission word.

It indicates to the host that the transmission from the 5216 is complete.

It may be noted, however, that the user is not supposed to send a ECT command for any reason. Inclusion of ECT word at the end of the message is automatically handled by the firmware.

2.13 USER PROGRAMABILITY

A block of RAM in 5216 may be allocated in which sequences or commands can be stored by the user. These sequences are then executed by a single command.

There are 2 buffers associated with the 5216 firmware namely, the Keyboard Buffer and the Cache Buffer. The Cache buffer must be loaded from the host but may be executed from the host or the Keyboard. The Keyboard buffer is more flexible than the Cache buffer. It can be loaded or executed either from the host or from the Keyboard.

2.14 ERROR CODE (ERC)

Whenever an error condition is generated, the display generator transmits to the host the Error Word, ERC. The code is defined only when following kinds of errors are there.

1. Host/5216 Communication Error

2. Cursor out of Limits

3. Illegal (Syntax) instruction

It is undefined for the rest of the code.

Chapter III

PSEUDO COLOUR FUNDAMENTALS

3.1 INTRODUCTION

In the modern age of artificial Intelligence, we have a rapidly expanding field in the area of Image Processing both in Black & White and Colour Images. In a manufacturing environment, eg. one would like a Robot to remove a defective piece on an assembly line and thus ensure good Quality Control of the finished products. For doing so, it is necessary to train the machine to distinguish between what is a bad job and what is a good job.

For a human observer it is of great importance and help if the Image is in colour. The reason for this is that we can distinguish only a few shades[9] in a grey scale image where as in a colour image the range of colour distinction is much better.[10] Thus it is worth while to investigate if a grey colour image can be translated into some form of colour

---

[9] The eye is only capable of judging the absolute brightness of only 10-15 shades of grey, but is much more sensitive to the difference in brightness of the adjecent grey shades [12]. See Photo [A.3.1].

[10] The eye can see thousands of shades of colour and intensities [1,10].

with some degree of success.

This is of great importance in Bio-Medical images where, for instance, a Red colour lump in a blue background could be identified as a Tumour by the Doctor. It is found that the features in an image come out more sharply if the image is assigned some colours instead of having different levels of grey [9,10].

The basic characterstic of the human vision is its ability to adapt to a wide variety of intensity and colour information.

In this chapter a brief discussion on the human vision and image enhancement is presented. This is followed by an introduction to Pseudo colour image processing.

Digital Image Processing has Applications in a number of areas such as medicine, Biology, Astronomy, Industry etc. All these fields share the common need for a method of enhancing the pictorial information for human interpretation and analysis. In medicine for instance, physicians could be assisted by computer procedures that may enhance the contrast or code the intensity levels into some form of colour for easier interpretation and understanding of the given image or an X ray picture [13]. Image enhancement and restora-

tion has been applied on several occasions when methods were needed to process degraded images depicting unrecoverable objects or experimental work too expensive to duplicate. Digital Image processing is used for machine perception also. Machine perception refers to extraction of information from images in a form suitable for computer processing. Typical problems in machine perception employing image processing techniques can be found in automatic character recognition, Industrial robots for product assembly and inspection and finger print identification etc.

3.2 THE NEED FOR A DIGITAL IMAGE

Continious tone pictures cannot be directly used for computer processing or analysis. This is due to the fact that the computers work on digital data rather than pictorial information. Due to this reason it is necessary to convert any given image into a numerical form before processing it. Another reason why an image needs to be digitized is because the storage of image in the memory would not be possible by other means.

A digital image is an image that has been digitized both in spatial coordinates as well as in brightness. A digital image, thus, can be denoted as F(X,Y,I) where X and Y refer to the coordinates of a pixel and I refers to its

brightness at that point. A digital image can be considered as a matrix whose row and column addresses identify a point in the image and the corresponding matrix value identifies the grey level at that point.

A continuous image can be approximated by a digital image having equally spaced samples arranged in the form of N * N array, where each element of this array is a discrete quantity.

This quantization process requires that a limit be imposed on the number of grey levels a picture can possess at the maximum. Also a decision is to be made regarding the size of such an array. It is usual in Image processing applications to let both these quantities be the integer power of 2.

$$N = 2^{n}$$

$$G = 2^{m}$$

Where 'G' refers to the maximum number of grey levels that can be allowed in a given picture.

From here it is easy to determine the amount of memory that would be needed for a given image. The total number

or samples that an image would have will be N * N. The variable 'm' determines the number or bits that have to be allocated to each pixel. This number can be varied depending the upon application but should be at least 5 for a black and white image and at least 8 for a coloured image for good perception. (See discussion in Section 1) However, it is obvious that the resolution of the digitized image will be closely linked to both m and N. A final choice will thus depend largely on the application that the user has in mind for the setup.

3.3 COLOUR FUNDAMENTALS

Light is the source of colour was first demonstrated by Isaac Newton, who passed a beam of sunlight through a glass prism, producing a rainbow array of colours of the visible spectrum. He passed this colour rainbow through a second prism, which reconstituted the original white beam of sunlight. This proved that colour is in the light and the light that we see is really a mixture of all the colours in the spectrum.

This phenomena of producing colours by the addition of primary colours was known to painters for a long time although it was thought that this is true only of the pig-

ments. It was only after the discovery by Newton that Colourimetry and the theory of colour were given a firm scientific basis.

Most colours can be formed by the additive or the subtractive comboinations of the primary colours. The primary colours for additive combination are Red, Green and Blue and are Cyan, Magenta and Yellow for subtractive combination. It is possible to determine the combinations of the primary colour that are needed to produce a specific colour from the chromaticity diagram. This diagram is a standard chart developed by the Commission on Illumination. It contains information on the Hue, Saturation and the primary colour components of any colour [12]. Hue refers to the actual appearence of, the colour to the eye and Saturation tells how deep the colour is. All the combinations that can be produced by any two colours lie on the line joining the two colours on the chromaticity diagram.

3.4 PSEUDO COLOUR PROCESSING

A recent development in digital image processing is the use of pseudo colour for image enhancement. The principle or the technique is to assign different colour levels to pixels having different intensity. It has been observed

from some experiments that the pixel that remain relatively inconspicuous in an image on a monochrome display become more apparent when the image is colour coded and displayed on a colour monitor [10]. From here it can be inferred that the pseudo colour technique increases the discrimination of the different pixels in a given image.

Since most scenes that the human eye perceives contain both intensity and colour information, it is only natural that we try to develop some technique through which colour may also be input to the eye from the computer. One can well imagine a number of areas where colour enhancement could be of significant value (eg. biomedical images and

seismology). The techniques discussed here convert the grey levels into different colours and hence the name Pseudo Colour.

It is to be clearly understood, however, that these colours that are artificially assigned to the image bear no resemblance to the original colours of the object. However Pseudo colours that have some resemblance to our senses can be produced on the screen eg. if a Thermal scan picture of a house is taken and that image is artificially enhanced with the warmer regions depicted in red and the cooler areas represented in blue then there will be some compatability between the displayed image and the notion about warm and cool colours.

The pseudo colour processing is initiated with a monochrome image and is mapped through a colour look up table and then projected on the screen to yield a coloured image.

There are a few techniques of converting monochrome images to Pseudo colour images, for instance Density Slicing and by performing three independent transformations of the grey level image into Red, Green and Blue images and then producing a composite image on the screen [1].

3.5 DENSITY SLICING

This technique in literature is sometimes also referred to as Intensity Slicing. This way of generating pseudo colours on the screen is arbitrary in nature and is seldom used in practice. In this technique a threshold is chosen and all pixels above this threshold are assigned one colour and all pixels below this value are assigned another colour as shown in fig. [3.1]. The pixels that lie on the plane itself may be assigned to any one of the two colours. The resulting image thus is a two colour image whose relative appearance may be changed by varying the value of threshold. Figure [3.2] shows the mapping function for this technique.

This technique can be further extended by having a number of planes intersecting the grey levels at different points. In this case the processed image will have N+1 colours, assuming that N planes are used. In the figure shown the image will have three colours.

3.6 GREY LEVEL TO COLOUR TRANSFORMATION

This utilizes the idea of additive properties of the three primary colours i.e. Red, Green and Blue. These colours are known as Primary colours because when added to each other in different proportions they yield almost all the

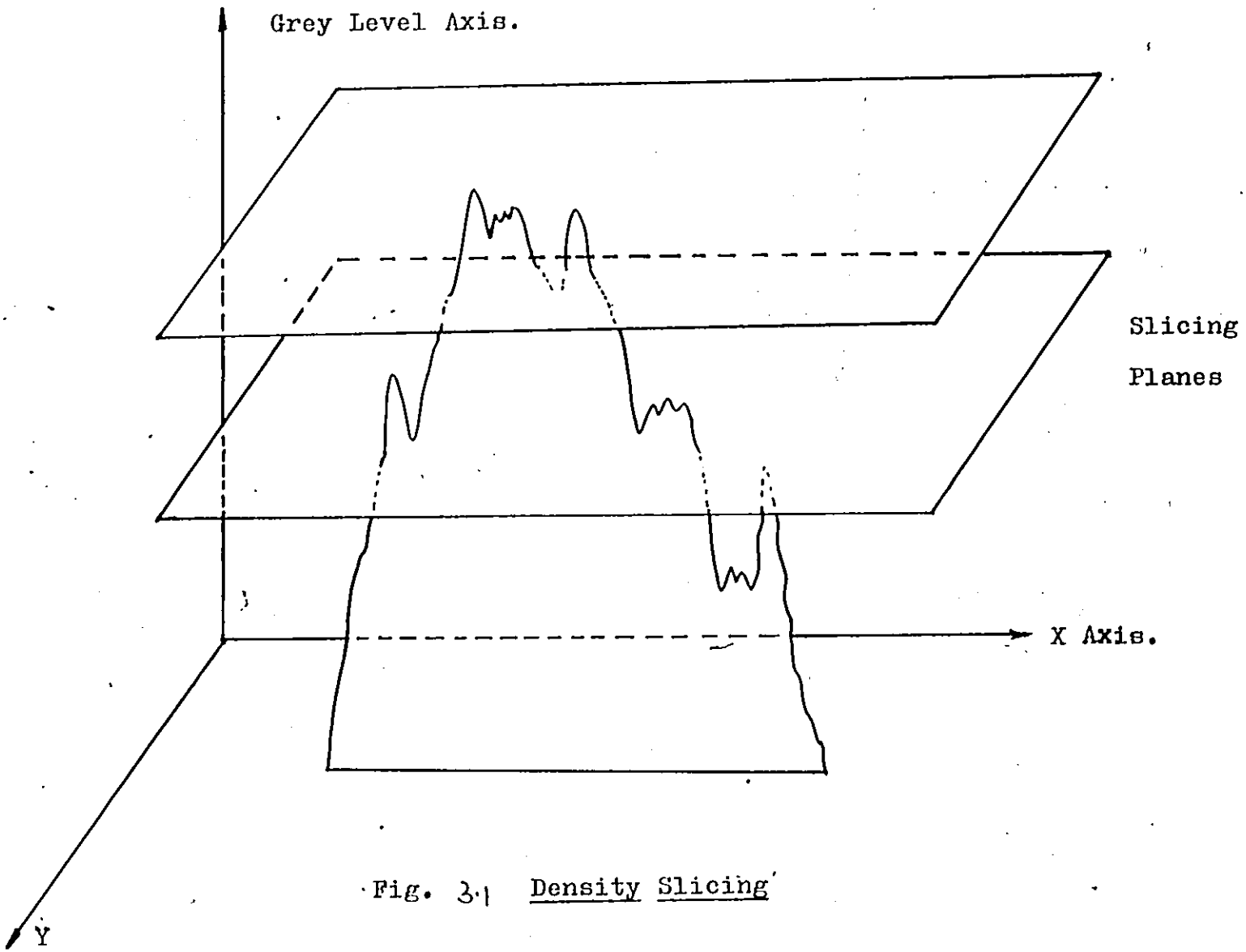


Fig. 3.1 Density Slicing

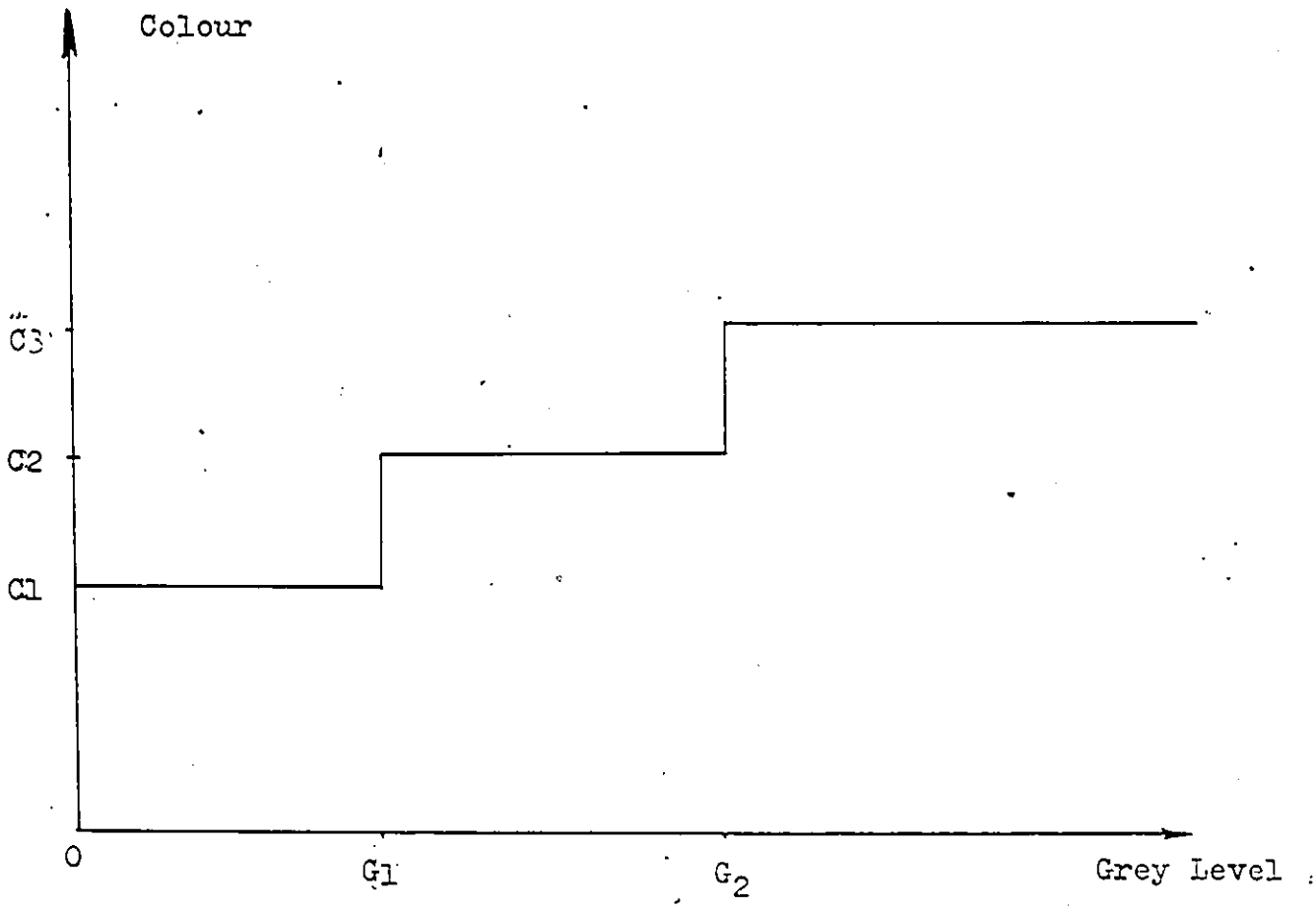


Fig. 3.2 Mapping Function for Density Slicing.

colours found in nature. When added in equal proportions, they yield white colour.

The second scheme is very attractive because of its easy implementation. The approach shown in fig. [3.3] could be used.

This technique is more general in nature and is capable of generating more colours on the screen. We can represent 3 colour intensities by Ig (X,Y), Ir (X,Y) and Ib (X,Y) where subscipts g, r and b refer to green, red and blue respectively. The variables X and Y refer to the position of the pixel.

3.7 COLOUR PRODUCTION ON AYDIN COLOUR MONITOR

In the present setup 256 distinct colours can be generated. The grey level data is 8 bits wide and the lower 3 bits drive the Red gun, the next 3 bits drive the Green gun and the last 2 are connected to the Blue gun. It is clear from here that we have 8 different intensity levels for Red and Green and only 4 levels for Blue. Since the present system is limited to 8 bits, only 256 different colours could be produced on the screen. It is a limitation imposed by the device. If a larger data width was available more shades of colours could have been produced.

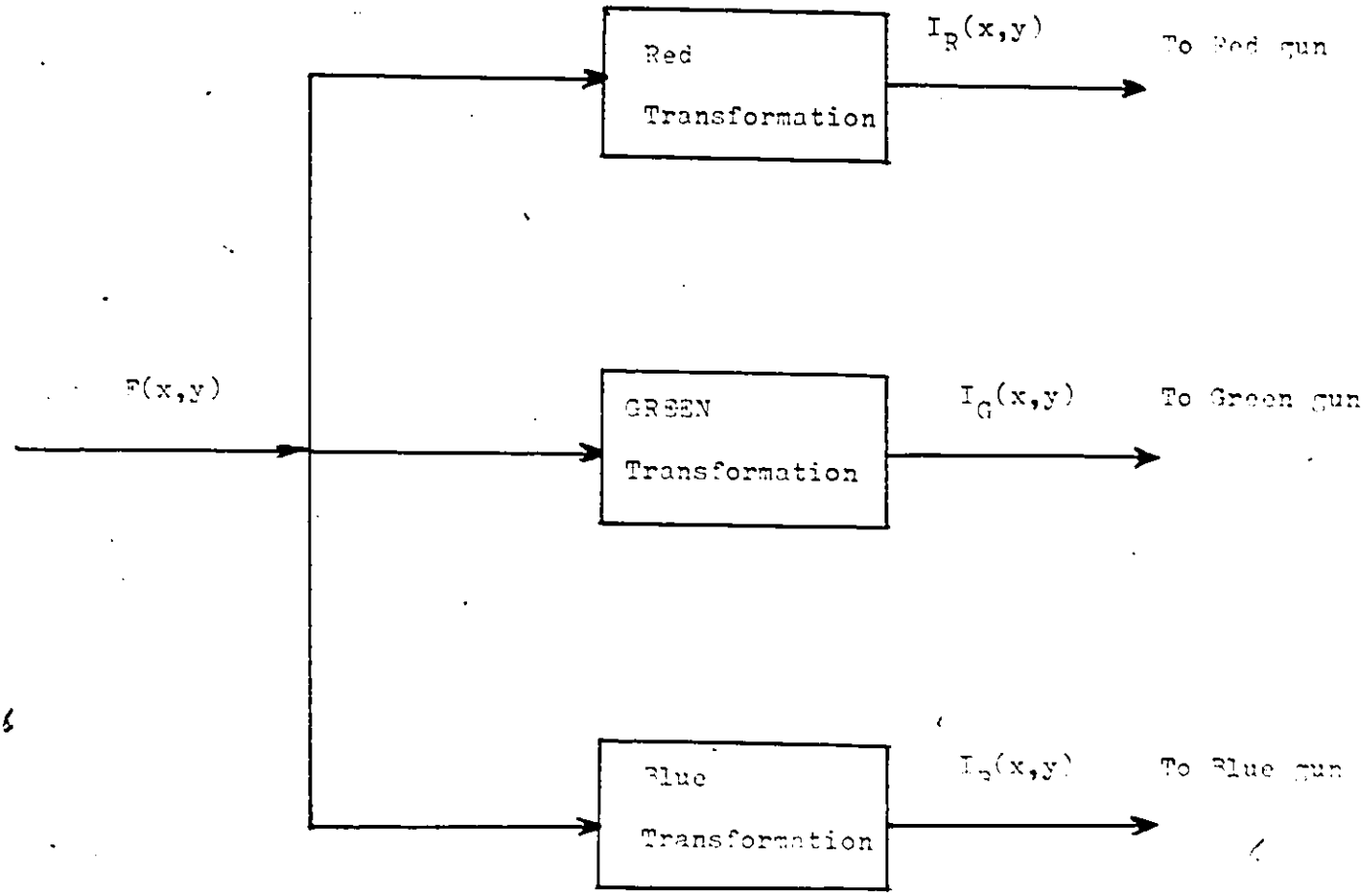


Fig. 3.3

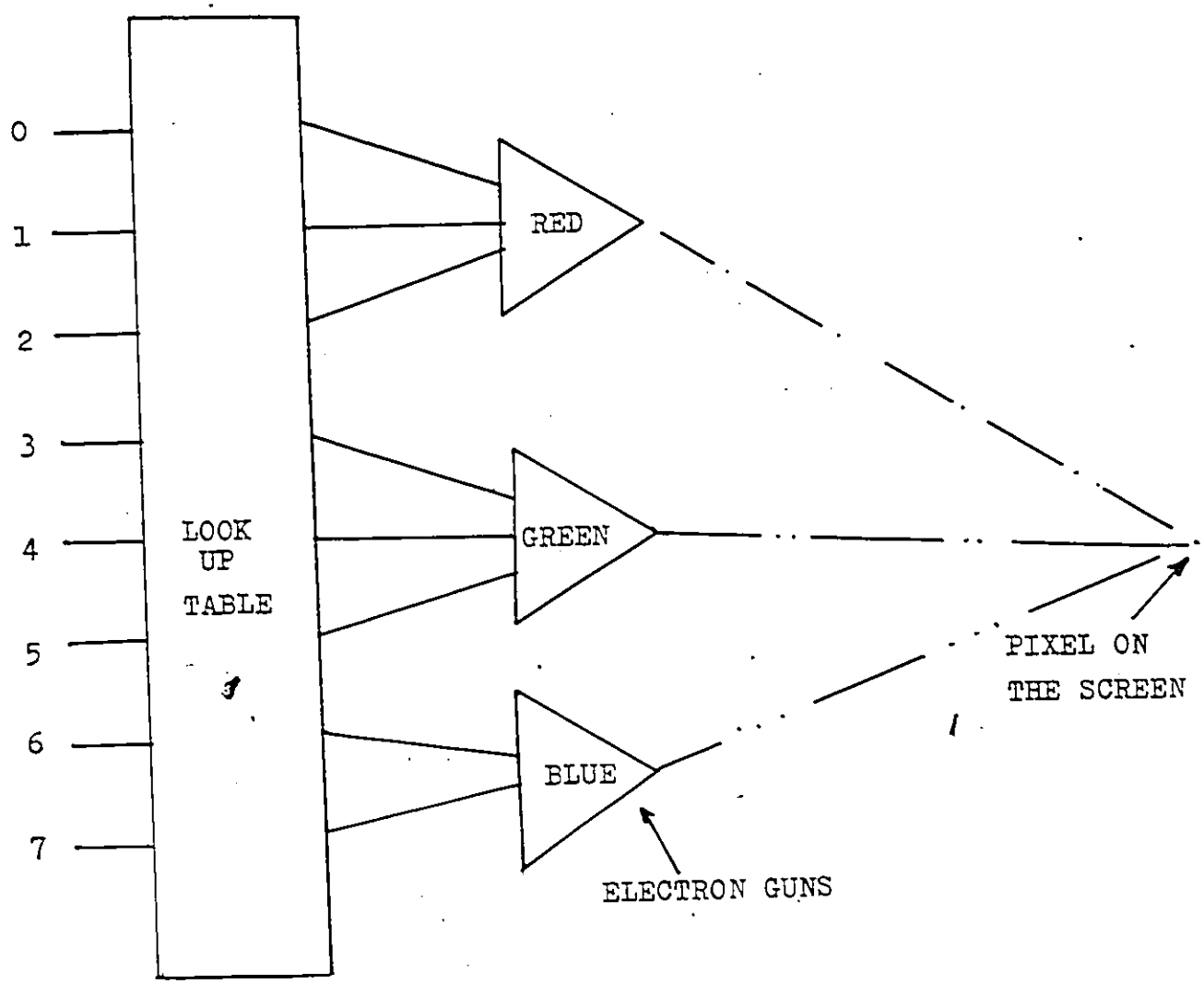
Block diagram of Grey level to Colour Transformation.

The results are fed separately to the Red, Green and Blue guns of a colour monitor [3] as shown in Fig [3.4]. Since all the three guns point to the same pixel at a time,[11] an overall composite image is produced on the screen which is the additive combination of these three primary colours. In this setup the transformation function is actually a lookup Table. Thus by changing the lookup table an entirely different colour can be produced on the screen for the same pixel values. It should, however, be noted that these transformations are on the grey level of the image and are independent of its position.

Two lookup tables have been provided in the Firmware namely, Lookup table # 1 (LUT # 1) and Lookup table # 2 (LUT # 2). Also the user is free to choose a lookup table of his own choice. Presently a lookup table that produces near real colours is loaded and can be activated by executing the program DISPLCOL from the Host computer See Photo [A.3.2, A.3.3].

---

[11] Each pixel is composed of 3 spots which are coated by electron sensitive phospher layer. Each of the phospher spots produces one of the three colours on the screen when excited by electrons. The effect viewed on the screen of the monitor, is that the three primary colours from each phospher triad are 'added' together and received by eye as a full colour image. These are known as Colour Triads.



BIT :	7	6	5	4	3	2	1	0
	B	B	G	G	G	R	R	R

FIGURE 3.4 COLOUR PRODUCTION ON AYDIN MONITOR

3.8 COMMUNICATION WITH AYDIN MONITOR

There are several ways to communicate with the Aydin monitor [2]. The three most practical ones are discussed here in some detail.

1. Monitor Keyboard.

One can use the monitor keyboard (5115) to edit the images on the screen. This is a very slow method and hence it is not a very useful way of editing the images. However, there is an option which when set can keep recording the keystrokes that one makes. Later these keystrokes can be retrieved. This is of interest if one wants to keep repeating the same operation over and over and one does not have any access to a more sophisticated way.

2. Cache Buffer.

The user can edit images on the screen by writing machine code instructions in the memory area of the Aydin monitor. These instructions can then be executed by a single command. The location of the instructions can be any where in the 1 Mega-Byte addressing space of the Aydin Monitor. There are 2 buffers, namely the Cache Buffer and the Keyboard Buffer. Both of them essentially operate in the same

way, but the difference lies in executing them as explained next. The Keyboard Buffer can be loaded or executed from either the host or the terminal's keyboard where as the Cache Buffer can only be loaded or executed from the host. The instructions are sequentially executed until a terminal word is encountered.

It is in this way that various images are displayed on the screen. It is not, however, a very sophisticated method. It is highly error prone since one has to deal with machine codes . It is also more difficult to write the code. There are problems in the maintenance, readability and upgradation of the program.

3. Fortran Package.

The most elegant way of communication with Aydin is by using the Fortran Support Package subroutines [3]. It is the most efficient way of editing the Pictures and is devoid of all the problems associated with the methods discussed earlier. A major Advantage in using the Fortran Subroutine Package is its Interactive ability.

3.9 SUMMARY

In this chapter a brief explanation has been provided for the Human visual system. The need for digitizing an image is then explained. A digital image is shown to be a 2 Dimensional array of numbers where X and Y refer to the position of the pixel and the quantity at that point represents the brightness of the pixel.

It is also shown that almost all the colours can be produced by the additive or subtractive combinations of the primary colours. The various combinations of the colours that can be formed by the addition of any of the three primary colours can be found from the chromaticity diagram.

The reasons for the development of Pseudo colour image processing techniques have been analysed. Pseudo colours increase the discrimination of different pixels in a given image. A technique to produce Pseudo coloured images from monochrome images has also been explained. The colours on the screen are produced by the additive combination of the three primary colours viz. Red, Green and Blue. This is followed by the explanation of the nature of colour production on the Aydin monitor. It is shown that The monitor has 4 different levels for Blue colour and 8 different intensity

levels for the Red and the Green colours. The total number of the colours are thus 256.

Communication between Aydin and the host SEL computer has been discussed. There are three different ways to communicate with the Aydin and they are through the monitors keyboard, Cache buffer and the Fortran package. Out of these the Fortran package is shown to be the most effective.

Chapter IV

WORK ACCOMPLISHED

4.1 GRAPHIC SOFTWARE

Along with the Aydin Display terminal, a Fortran Software Package to process the images was also received. As a part of this work this software package has been suitably modified and implemented on the SEL system.The package interprets the call given for each subroutine. It then appropriately generates and outputs one or more 16 bit commands to the colour monitor.

The functions supported by these subroutines are the same as the ones provided on the keyboard of the Aydin monitor namely Zoom, Fill, Copy, Drawing conics and vectors, setting and resetting of rectangular and conic viewpoint limits, positioning the cursor etc. There are in addition several subroutines which do certain jobs which are not available on the keyboards: for instance, Executing and loading the cache buffer and switching the visible cursor on or off. The basic advantage in using the package is that it is a quick way of doing the job reliably, and with a lot of ease.

These subroutines in the original package were written for a 1024 pixel square screen. As a part of this study this subroutine package is modified and implemented on the 512 pixel squared screen system based around the host SEL computer.

In the subroutine package provided, a few routines did not work at all. After a complete examination and understanding of the package it was found that the subroutine BLOCK DATA INIT was the root cause of most troubles. This routine is responsible for initialising the various parameters that are used in this package. Some other routines which did not work were the ones which roll or scroll the data nondestructively etc. Adequate changes are made in these routines by inserting statements in each of them. The insertion of statements is done in order to make the programs more general in nature. These statements instruct the routines about the screen size and other parameters of the system. These modifications are carried out with a view that if the user wishes to transplant the package to any other screen size, the routines would run perfectly provided the screen size is intimated to the routine INIT. Every other subroutine would automatically be instructed of the change. INIT is a Block data subroutine which is meant to initialise the parameters for use in all other routines. Every time the program runs into a parameter whose value is

nct specified in the main program, it is searched for in this routine. It should be noted, however, that all this is actually transparent to the user.

Subroutine IOINIT (that is standardised for use in the programs) does the job of communicating with MPCI board, enabling Break receivers, Assigning Logical File Codes (LFC) and opening up the channel and initialising the MPCI.

A few other problems associated with these routines were also taken care of and now the whole package works correctly.

The method to use this package is explained through fig. [4.1]. A default length for all the variables is established at INTEGER*2. This is done in order to cut down on the memory requirement in the programs. A few parameters, however, need longer word lengths and are provided accordingly. All the programs which have to be listed as external to the library are listed under the EXTERNAL statement. Subroutine INIT although a part of the library has to be listed as an External routine. This is a Fortran requirement [7].

FLOW CHART OF A TYPICAL PROGRAM CALL

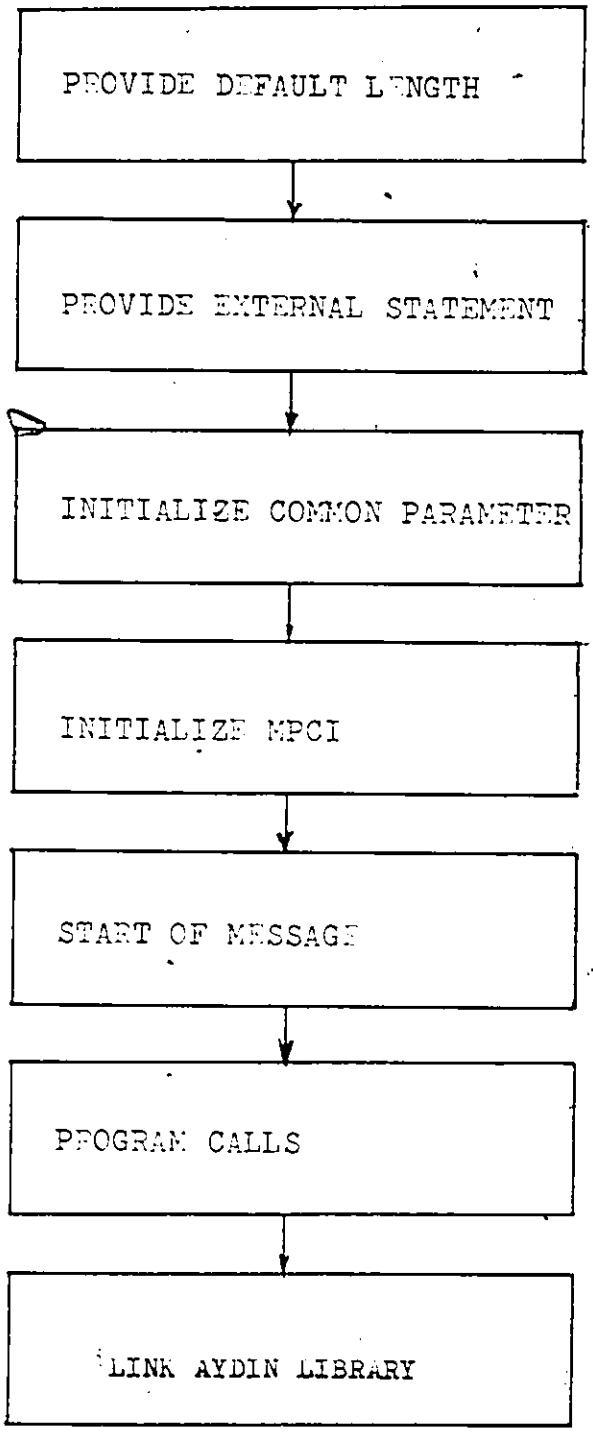


Fig. 4.1

Next the MPCI is initialised for operation. This call readies the MPCI card to open up the communication between the host and the device as explained elsewhere. At this point of time the display system has been accessed. In order to start writing any thing on the display monitor, however, a Start of Message command is to be given as explained in chapter 2. This command orders a specific parameter set to be loaded into the Aydin monitor. In most cases parameter set # 1 is used in the programs. One may, if needed, associate different parameter sets to different cursor. Once a particular set is chosen then the user can insert any number of calls to the package's routines.

At this stage, it must be noted that the user has to link the Aydin subroutine package to the main program. This is done in the job cards following the main body of the program. The cards to be inserted are as follows.

```
$ASSIGN LIB TO @DP0 (SYSTEM) AYDLIB BLOC=N

$ASSIGN DIR TO @DP0 (SYSTEM) AYDDIR BLOC=N
```

This program is then compiled and run as one of the usual Fortran programs.

4.2 SELECTIVE ZOOMING

In some cases it is important for the user to examine a small part of the screen for closer details eg. for locating any hair line cracks. A 128 * 128 sized image may be small for this purpose and there is a need to zoom it. Since only a small part of the image is to be observed it is only necessary to zoom that portion of the image.

Such a facility has been developed on the SEI computer. This could be used by calling the program SELZOOM. After the image is displayed the user decides that a small portion of the image is to be examined. The user touches a part of that image and with the touch position as the center a 32 * 32 image is copied to another part of the screen and is zoomed by a factor of 2. A square is also drawn around the touch position to identify with the area that has been zoomed. The main strength of the program lies in the fact that the user need not specify any coordinates. This process can be repeated any number of times. In order to exit from the program the top right corner of the screen should be touched.

4.3 HISTOGRAM PLOTTING

As discussed earlier, Histogram plotting is an important way of analysing if the pictures grey levels are distributed or are clustered together on some value.

A program to draw the histogram of any image stored in the memory has been developed. This program is written for a 128 * 128 sized image, though it can be easily upgraded for images of other sizes.The program after calculating the histogram also smoothens it so that the valleys and the peaks may be observed more clearly.

Any image that is stored in the memory can be plotted for its histogram by this program provided, it is first assigned as follows.

```
AS 1 TO @SYSTEM(IMAGE)[ Imagename ]
```

[Imagename] is the name of the image for which the histogram needs to be plotted. After this assignment the program is run. The compiled version of type program is named HISTSV. Photo [A.3.4] shows a histogram plot drawn on the screen.

The hard copy of the same may be obtained on the printer [11] .

4.4 SEL EXECUTIVE

The next part of this work involves fitting these subroutines into the SEL executive.

The scheme is illustrated in the accompanying fig. [4.2]. A choice of menus i.e. Display system, Printer, Convolver, Camera etc first appears on the screen. After a specific operation is selected the first level or submenus appear on the screen i.e. in our case it is a choice between operations, displaying of an image or reverting back to the main menu.

In the case of the display system menu, it was decided that only a few of the important routines be kept in the menu. It is necessary because of the fact that there are a very large number of routines. To keep the number of menu pages small, a limit on the number of routines on the touch screen is placed. Programs e.g. Zoom, Copy, Drawing of Vectors and Circles, Clear screen, rolling and scrolling of images non destructively have been put in the menu.

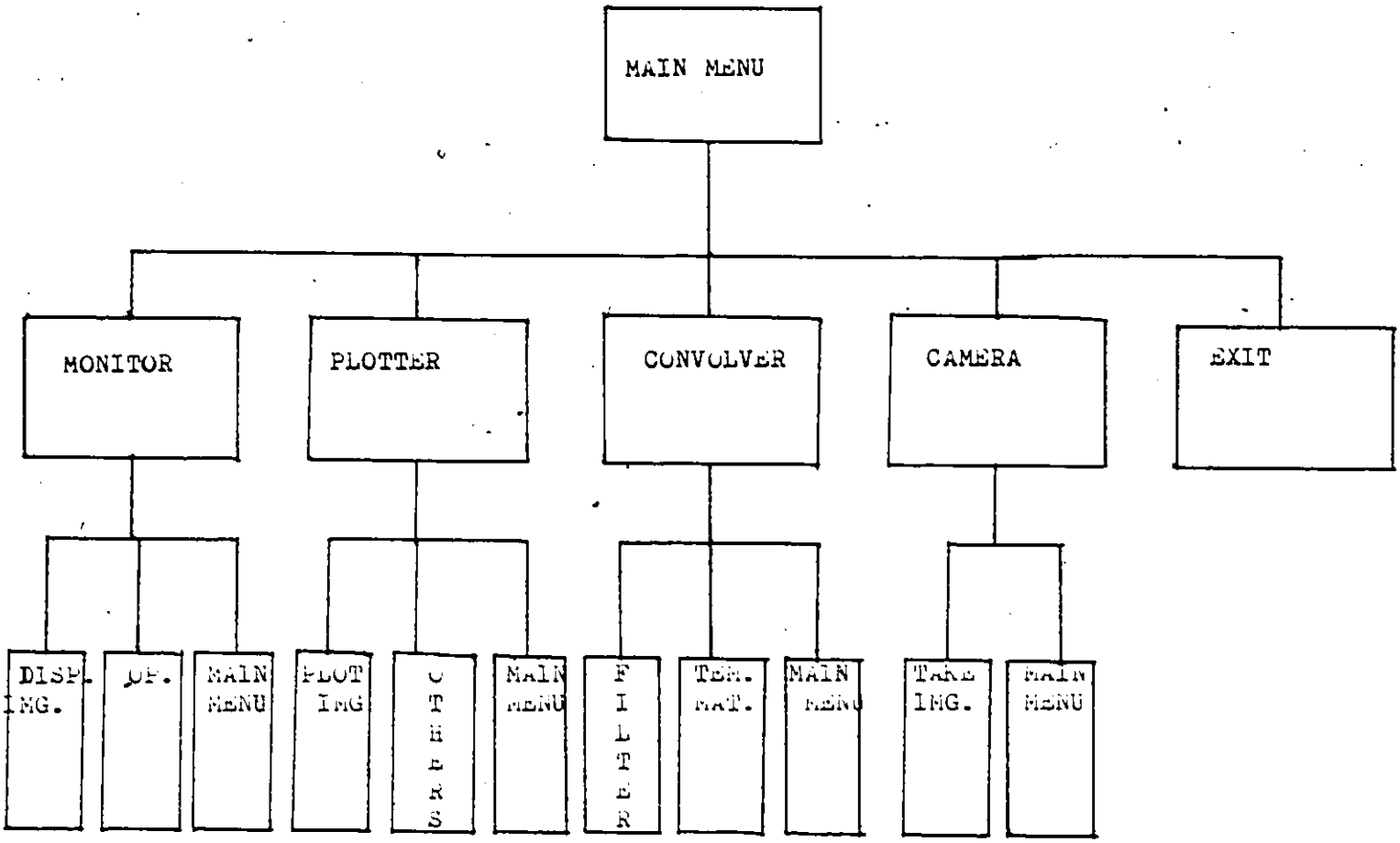


Fig. 4.2

Inverted Tree Structure. This is a common method for providing easy acess to a wide range of pages. One proceeds through a series of pages until the desired menu is reached.

Subroutines such as cursor position readback, loading the look up table, writing programmable characters, start of message, select device, font size changing etc. have been deliberately kept out of the menu because of two reasons - firstly they are seldom used and secondly because they would have made the menu very difficult to use.

If from the main menu, the display system submenu is selected, it will appear as shown in fig. [4.3]. The Numbers in brackets are the coordinates on the screen. The menu as it appears on the screen is shown in Photo [A.3.5].

In the following section, the way to use each of these routines is explained.

4.4.1 COPY IMAGE

As explained elsewhere in the thesis, the copy command, prior to getting executed needs that the cursor be positioned in the center of the image section to be copied. The area around the cursor will be copied to the section defined by the conic limits. So in order that this instruction be executed, it is necessary that these variables be defined before hand.

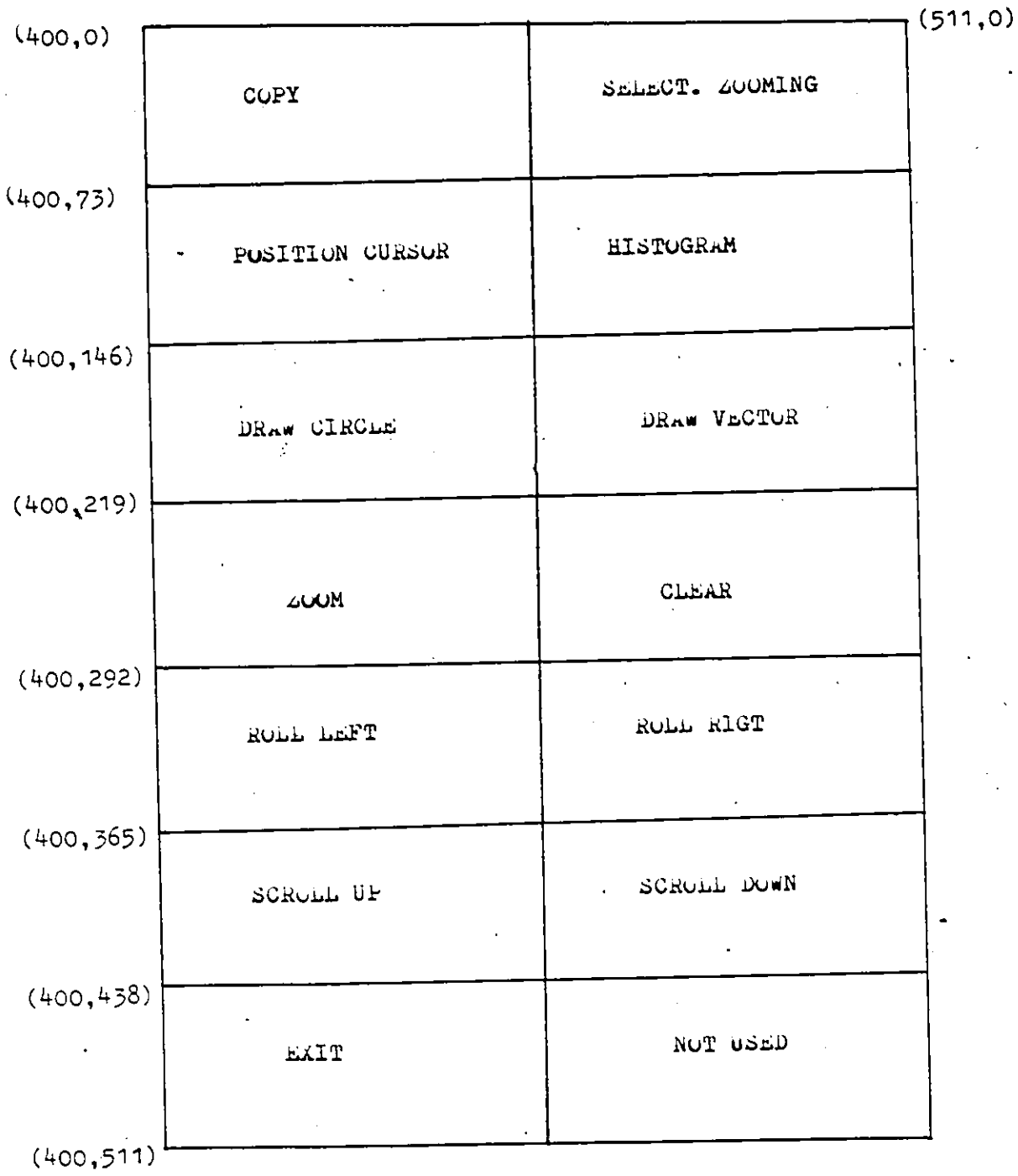


Fig. 4.3

Display Menu Configration.

The method to use the routine is as follows:

After touching the screen for copying the image, the user touches the image center, to position the cursor there. Then the conic limits are defined by touching the top left corner and the bottom right corner of the conic limits in any order. After this is accepted the image will be copied to the area that has been defined by the limits.

4.4.2 SELECTIVE ZOOMING

Once this instruction is executed by way of touching the soft switch generated on the screen, any image when touched is zoomed by a factor of 2 around the touch position. In order that the original image remain intact, the part to be zoomed is copied to another part of the screen and then zoomed. To exit press soft switch named EXIT.

4.4.3 POSITION CURSOR

By touching the menu button, any portion of the screen can be touched and the cursor will be positioned at that point. It should be noted that the cursor may not be placed outside the rectangular limits.

4.4.4 DRAW CIRCLE

Circle drawing basically needs two inputs. One of them is the center and the other one is the radius. Both of them are provided to the screen by the touch position in this setup. First the user touches the center of the circle and the second touch position is measured as the radius. Based on these inputs the circle is drawn.

This routine has been further developed so that concentric circles can be drawn on the screen. The first two touch are associated with the center and the radius of the circle. If now the user wants to draw another circle with the same center all that is needed is another touch. The need to specify the center has been done away with. To exit the circle drawing routine touch switch EXIT.

4.4.5 DRAW VECTOR

It operates in essentially the same way as above. The first touch position is interpreted as the starting point of the vector and the second touch as the end point and then the vector is drawn.

Here also the first 2 touch positions are interpreted as the starting and the end points. A closed figure can be

drawn by touching the vertices of the figure. The end point of one vector becomes the starting point of the next. To exit from this again EXIT may be touched.

4.4.6 ZOOM

Any image may be zoomed by a factor of 2 by using this program. In order that this program is completely matched with our menu organisation it was decided that first the image needs to be copied to another section and then zoomed. It was necessary because whenever the images are displayed on the screen through the touch screen they get displayed on the top left corner. If they are zoomed right there we will only be able to see a part of the zoomed picture and the rest of it will be lost.

4.4.7 CLEAR

Touching of this soft switch will clear the screen to 0. The menu will, however, not be cleared from its position.

4.4.8 ROLL LEFT

A touch on this switch will roll the image left by 5 pixels.

4.4.9 ROLL RIGHT

A touch on this switch will roll the image right by 5 pixels.

4.4.10 SCROLL DOWN

A touch on this switch will scroll the image down by 5 pixels.

4.4.11 SCROLL UP

A touch on this switch will scroll the image up by 5 pixels.

4.4.12 HISTOGRAM

This menu button, when pressed, processes and displays the histogram of the piston head.

4.4.13 FILL

Any closed area can be filled with a colour through this soft switch. If the area is not closed, the colour will spill to the whole screen. After a touch on this button only the area to be filled with colour needs to be touched at any

point inside this area. A choice of colours has not been provided to the user because this would unnecessarily make the menu more complicated both in terms of usage and programming. The speed of operation will also be adversely affected.

The menu will be permanently appearing on the screen, unless the user wishes otherwise. Because of this reason the whole of 512*512 pixel squared screen will not be available to the user. A small part of the screen has been reserved for the use with the touch screen. The rest of the screen will behave normally. Even if the user gives a clear command the menu will not be disturbed from its place, only the picture will disappear from the terminal. If, however, the user needs to clear up the whole screen for some reason, a command has to be provided from the keyboard resetting the full screen limits and then it can be cleared by giving a Clear command either from the touch screen or from the keyboard. In other words, for most practical purposes the screen size is limited to only a part of the whole screen. Any data pushed out of this boundary by way of zooming, rolling or scrolling the image will be permanently lost.

Now with the given system the user will just have to decide the function needed and the appropriate soft switch on the screen to be pressed. The function is then performed automatically.

4.5 COLOUR MATCHING BETWEEN AYDIN AND PRINTER

Another aspect that is explored as a part of this thesis was that some colour matching needs to be done between the Trilog printer and the Aydin colour monitor. This is to be done because of the fact that Trilog colour printer can at best produce 64 distinct colours where as the colour monitor has the capacity to produce 256 distinct colours. Also the colours are produced on the terminal by the additive combination of the three primary colours viz. Red, Green and Blue whereas on the Trilog printer the colours are produced by the interference of the subtractive primary colours viz. Magenta, Cyan and Yellow.

Because of this problem, a scheme is develoed wherein, 256 colours from the Aydin terminal are effectively translated into 64 colours at the printer without much loss of colour match [11].

Another problem that is encountered in the data transfer between colour monitor and colour plotter is that the monitor can display 512 pixel square screen whereas the plotter can only print 128 pixel square image. This problem is resolved by transferring only 128 pixels of the data to the plotter at a time. A number of sections of the picture are then pasted to have a complete picture.

4.6 INTERACTIVE QUALITY CONTROL PROCESS

Usually to determine the quality of a batch, a scheme is employed wherein random samples are picked from the batch and examined for quality and the batch quality is projected from this sample through statistical means. As a part of this work a technique is proposed where job is checked for faults such as misalignment of gears etc.

In such a situation, ideally the systems camera should be able to picture the part. This picture is then digitized and stored in the memory. The computer then picks up this image from the memory and processes it to isolate some features of the image and then these processed images are displayed on the display monitor of the system.

The processing to be done could be thresholding of the stored image, its boundary detection and the histogram of the image under observation. These features thus extracted may then be compared readily to another good piece whose data is already stored in the memory.

Such a scheme has been developed on the SEL computer now. In order to use it, the user needs to run the program CNQCI along with the image name on which the processing needs to be done. eg. If the processing is to be done on

the transmission gear, the following command will have to be given.

CNQCI TRANS128

TRANS128 is the name of the transmission gear image of size 128.

Once the program is executed it displays the original image, its histogram, Thresholded image, its boundary etc. The facility to zoom a select portion of the original image has also been incorporated in to it. This has been included because if the operator wants to inspect a part in closer detail, this feature will be useful. A square is made around the touch position so that the user be able to identify with the original part of the picture. This process can be repeated any number of times. To exit from this program the User can touch the soft switch named EXIT that appears on the screen. Photo [A.3.6(a), A.3.6(b)] show the processing done on 2 Images.

The screen size is limited in our setup so the images of the objects for comparision can not be displayed simultaneously. However, if the screen size is larger it will not be a problem.

This kind of stations can be put on a number of places. The operator at any given station visually examines the data that is automatically presented on the screen mounted in front of him and decides if the part under inspection is to be passed or failed. In the event the part passes the quality control at that stage, it is allowed to proceed to the next stage, else it is taken off from the assembly line. For this purpose also, the screen can be used - where a part of the screen is touched which runs a program to activate a robot arm which will picks up the piece from the line and discard it.

Right now the processing is being done on the stored images only. This program can also be used for an ongoing process without any modifications. When this program is used for on Line inspection , it should be put in an endless loop, where the image is stored in some section of the memory, the results of this image are then presented to the operator, and after a decision is made, the next job on the assembly line is pictured by the camera and is stored in the same memory. The whole process is repeated again.

The processing will be highly dependent on the work environment. In a place where something else is to be checked for quality, the basic programming structure will remain the same, even if, the features that will have to be

isolated from the processing may be different. This program can be modified and tailored to the needs of a particular environment easily.

4.7 PRODUCTION OF REAL COLOURS ON THE MONITOR

Since most (not all) colours found in nature may be produced as a combination of 3 primary colours, it was decided that it was worth while to investigate if Real colours can be produced on the screen in this way.[12]

In order to produce real colours on the screen it is necessary to produce 3 different pictures of the same object through 3 different filters. The first picture is made with a red filter in front of the camera lens, intended to pass only the red portion of the image colours and absorb the rest of the visible spectrum. The second is made with a Blue filter on the lens, to pass only the blue light and the third with a green filter to pass only the green light. Each picture thus captured is still black and white but represents the amount of each colour in the given picture. All these 3 images are then stored in the computer memory. The

---

[12] There is only one way of producing "all" the colours as they appear in nature, but it is impractical to produce them. It was demonstrated by Prof. Gabreil Lippman, a Professor of Physics at Sorbonne, and it fetched a Nobel Prize for him in 1908. His process made use of the phenomenon of light interference, the interaction that produces the brilliant colours as seen on soap bubbles.

red, green and Blue components of the coloured image are as shown in the Photo [A.3.7].

In the absence of a working camera interface with the SEL computer these images are taken on the NOVA 840 computer with a C 1000 Hamamatsu camera. The images captured are 256*256 pixel squared . These images are then reduced to a resolution of 128*128 pixel squared each by reading only the alternate pixel row and column of the original image. These three images are stored as REDIMG, BLUEIMG and GRNIMG to identify which part of the spectrum each of them corresponds to. Care is taken to ensure that the images when taken, are taken with the same camera position, object position, aparture and focus. Any change in these parameters will make the final picture look blurred and the colour renderation may also be affected.

These pictures are transferred to the SEL 32/27 Machine through the interface between the 2 computers.

The next part is to produce a composite image from these 3 images. Theoretically a image having true colours can be produced from these 3 pictures by projecting the red image by red light, the green image by green light and the blue with the blue light source and then these images are precisely superimposed on each other to give the final ef-

fect. But on the Aydin terminal, all the 3 images can not be viewed on the same part of the screen simultaneously - so a scheme is developed by which these 3 images are combined into a single image and yet retaining the correspondence to the grey levels of the component images.

As mentioned earlier in the thesis, the 8 bits of the data available to the user is connected to the blue, red and the green guns of the monitor. The green and the red gun each is connected to 3 bits and the blue gun is connected only to 2 bits of the data. So the total number of the intensity levels for red and green are 8 each and 4 for blue.

The representation of the bit pattern is as follows:

7	6	5	4	3	2	1	0	Bit Number
B	B	G	G	G	R	R	R	

G = Green

R = Red

B = Blue

In the original images that we had captured through the Hamamatsu C 1000 camera, each image had a grey level re-

sclution of 256. Clearly this many grey levels can not be accomodated for each colour in the composite image that is to be constructed. So it was decided that the levels on the original images be scaled by some factor so that they are represented on the original image. The grey levels are scaled to 8 for red and green images and 4 for the blue image. Thus at this stage 3 arrays each scaled down to 4 or 8 levels of intensity as applicable in the individual case are formed.

These arrays are then combined to make a single array which produces the added effect of each of them. In order to do so, the blue array is read first, the two bits of data thus read is rotated by 3 bit positions. Next the green array is added to it. The lower three bit positions that were clear are then filled by the green image data. Again each of the array element is rotated by 3 bits and then the red file is added into it. The resulting data has blue data in the first 2 bits, green data in the next three bits and red data in the last 3 bit positions.

The next thing to be done is to load a look up table in the memory that enables these bits to pass through without any modifications on them. It is necessary to load a look up table because there is always one which is loaded in its memory by default. Since all the bits in the data are

at the right place no modifications are needed on them. Such a look up table has been written and loaded in the Aydin memory. It is necessary to load this lookup table in the Aydin monitor before the composite image is displayed. This program name is RELCOLS.

After this the image can be displayed on the screen. The name of the image is FNLIMG and can be displayed in the following manner.[13] The number of different colours on the screen are still 256.

```
A1 1=FNLIMG
DISPLAY
```

The original and the composite images are as shown in Photo [A.3.8] and [A.3.9] respectively. In [A.3.9] a number of variations were tried to get a close colour match. Some of these variations are shown in the photo.

---

[13] In order to display any image from the memory the following commands should be used.

```
AS 1 TO @SYSTEM(IMAGE)imagename
DISPLAY
```

One of the variations that can be tried out on this setup is that only the red and the green arrays are taken into the final image and the blue array is not taken into account.[14] This is done because blue is not a critical colour and may not be needed in all the images. This scheme can be used when the user wishes to save on the computation time.

The flow diagram of this scheme is represented as shown in fig. [4.4].

4.7.1 RESULTS OBTAINED

After the results were finally viewed on the Aydin colour monitor it was found that only the yellow colour has been reproduced faithfully. The other colours suffered from some distortion and blurring. Eg. The steel grey colours in the picture were converted to black and light blue colours also turned black. The white and black colours, however, are produced without any change.

---

[14] Earlier photographs were produced by the additive combinations of the primary colours. Since adjustment of the 3 colours was difficult, the photographers used only Red and Green components of the picture. This technique also gave reasonably good colours.

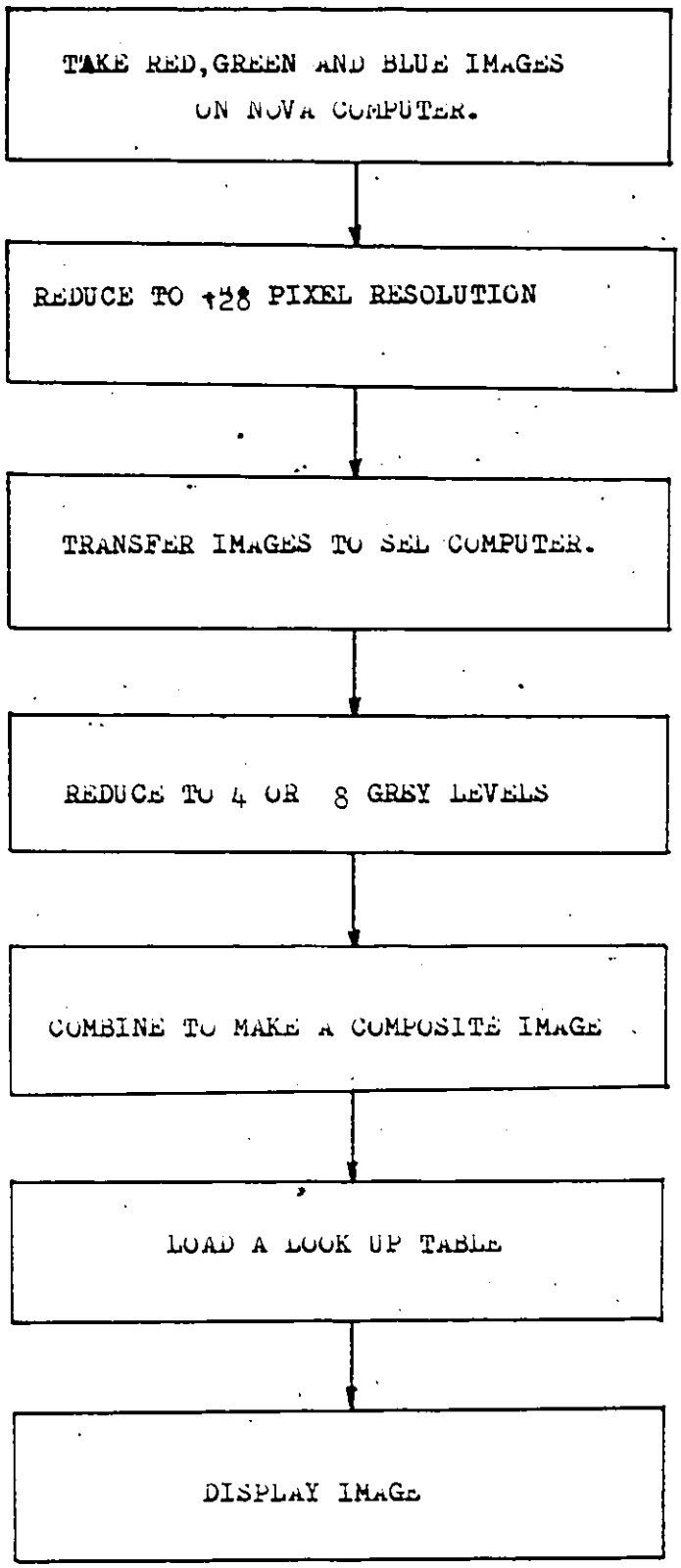


FIG. 4.4

FLOW DIAGRAM FOR PRODUCING REAL COLOURS

The reasons why the technique did not work as well as expected could be attributed primarily to the following factors. It is interesting to see that all these factors are device dependent and not on the technique.

1. Reduction of 256 colours into 4 or 8 intensity levels.

2. Pixel resolution of the picture.

3. The colour production on the Aydin monitor.

Each of these factors are now discussed in some detail.

4.7.1.1 REDUCTION IN THE INTENSITY LEVELS

Since blue colour could, at most, be allowed to have 4 different levels of intensities, a severe error has been introduced. Eg. level 0 and level 63 both map into 0 level in the new image. Similarly the red and the green images also suffer from the same problem. This error can not be avoided because of the nature of the colour production technique employed by Aydin.

4.7.1.2 PIXEL RESOLUTION

Since the pictures taken were 128 * 128 pixels, the resolution of the image is poor. This dimension of the picture can be increased but then the processing time to produce the composite image is quadrupled and the effect on the resolution is only minimal. So this factor though present is only minor as compared to the others.

4.7.1.3 AYDIN COLOUR PRODUCTION

The nature of the colour production on Aydin itself suffers from a few limitations. The colour changes are too abrupt on the monitor. Also when the intensity of the colour is low, the Aydin produces a black colour instead of a light shade of this colour.

CHAPTER V

CONCLUSIONS

A touch screen executive developed as a part of this work is a very 'user friendly' system to edit the images on the screen. This executive has been designed as an inverted tree structure which is a way of reaching a specific menu page in the least number of selections. The options are presented to the user in a predetermined fashion. The user responds to such a system by touching the soft switches that are generated on the screen. An effort has been made to keep the menu concise and efficient. The executive has been so designed that the user, any time the need arises, can add another page of the menu to the existing structure. Thus the upgrading of the system will not be a problem.

Software has been written to display images from the memory in Pseudo colours on the screen. The appearence of the image can be changed by loading a new lookup table in the Aydin memory.

The fortran package provided by the Aydin Display, Inc., to edit the images on the screen was modified to suit the setup based around the host SEL computer. A flexibility

has been added to the package so that if it is transplanted to another screen size, the only change needs to be made in the initializing routine BLOCKDATA INIT.

Software for histogram plotting and selective zooming has also been developed.

A software scheme for interactive quality control has been implemented. In this scheme the Original image, Thresholded image, its boundary and its histogram is presented on the screen automatically. The user is also presented with an option on the touch screen for user threshold selection or automatic threshold selection. If the former option is chosen the threshold level is specified by touching the histogram of the image displayed on the screen.

A scheme to produce colours close to the original colours of the object has also been implemented. From the results obtained it seems that these 'real colours' can not be faithfully produced on the Ayuin monitor. The possible reasons for the technique not working well can be attributed to the following reasons.

1. severe quantization is introduced due to the scaling and this seems to supress the colour information in the picture.

2. The filters used to capture the Red, Green and Blue images passed some portion of other colours eg. the red filter in addition to the red colour also admitted the green and the blue portions of the image. This led to the 'mixing' of the three colours.
3. The colour changes on the aydin monitor are too abrupt instead of blending into one another smoothly. When the intensity of any colour is low the monitor produces a black colour instead of a light shade of that colour. Also the colours in the range of 128-255 are predominately blue. So although the total number of colours that are available are 256, the red and green colour combinations are relatively low.

Due to these factors it is felt that the Real colour production on Aydin is not possible.

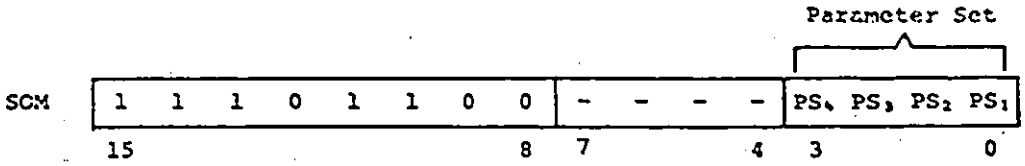
Furthur study can be made to establish what is the minimum number of bits that are needed to produce reasonably good real colours on the screen. A study can also be conducted to see what is the individual effect of each of the reasons cited above on the colour reproduction. As an extention of this work it is possible to enhance the touch screen executive to include other software developments in the department.

Appendix A

AYDIN FIRMWARE INSTRUCTIONS

OPERATION CODE SUMMARY

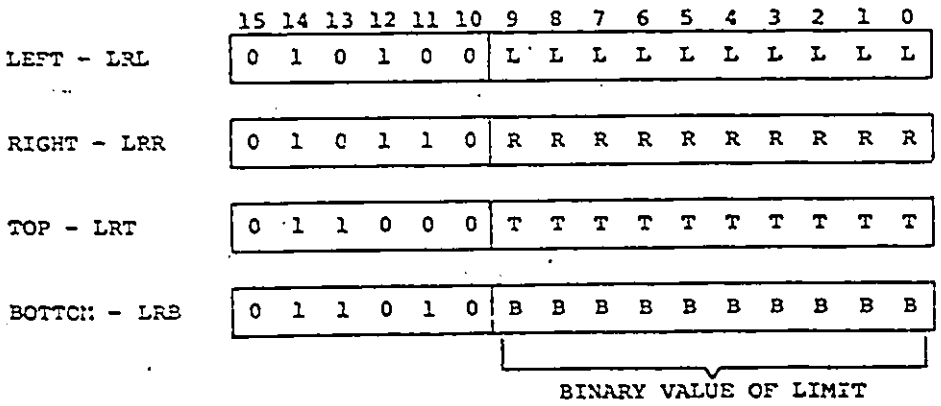
Start of Message - SOM



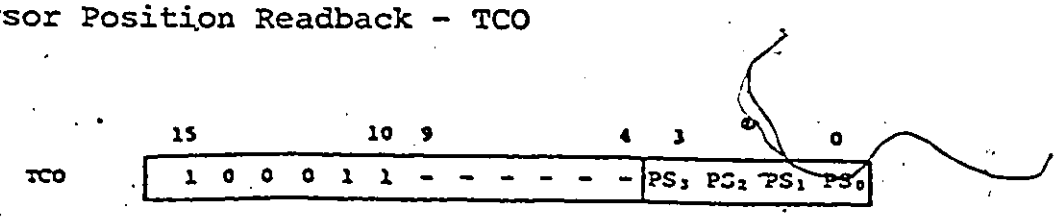
Parameters

- MAJOR Channel Selects
- MINOR Channel Select
- MAJOR Channel Mask
- MINOR Channel Mask
- Mode Control Word
- X - cursor position
- Y - cursor position
- X - index
- Y - index
- Foreground pixel value
- Background pixel value
- Rectangular limits
- Conic limits
- ACA values
- KTS

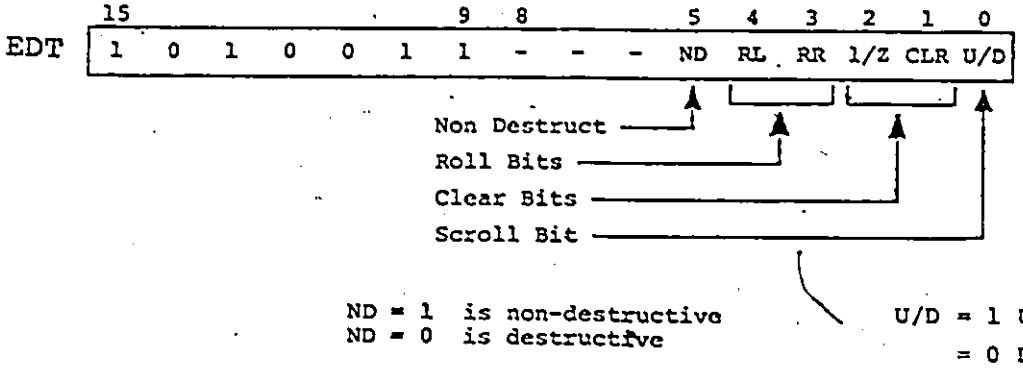
Load Rectangular Limits - LRR, LRL, LRT, LRB



Cursor Position Readback - TCO



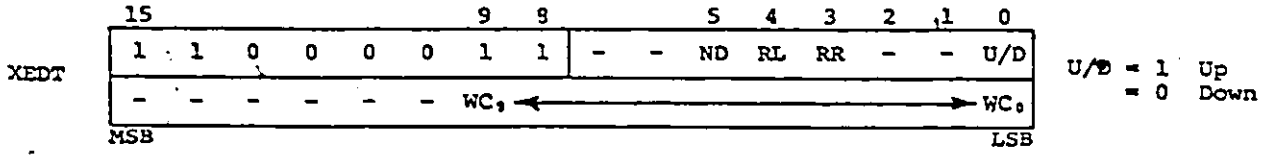
Scrolling, Rolling, Clearing - Edit Instruction EDT



Bit No. 2 1/Z	1 CLR	0 SCR	Meaning
0	0	0	Scroll Down
0	0	1	Scroll Up
0	1	0	Clear to Zeros
1	1	0	Clear to Ones

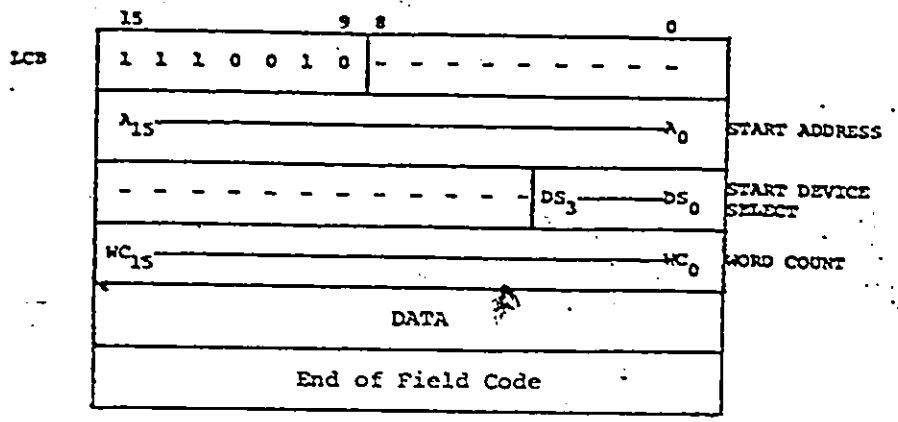
Bit 4	3	Meaning
0	1	Roll Right
1	0	Roll Left
1	1	Danger

Extended Edit - XEDT *EXT



End of Transmission - EOT

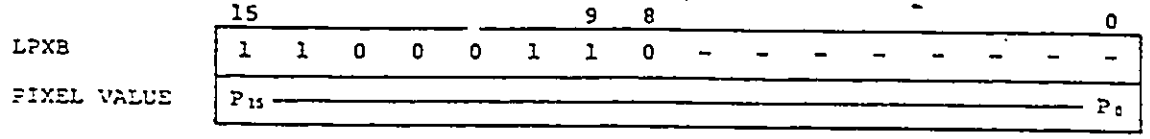
Load Cache Buffer - LCB *EXT



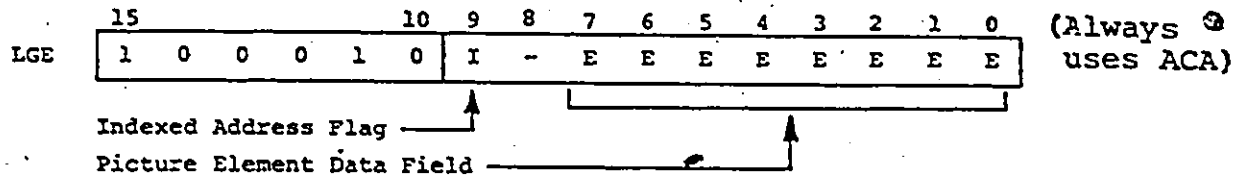
End of Field Code

= FC00 hex for 5216
= CB hex for 8086

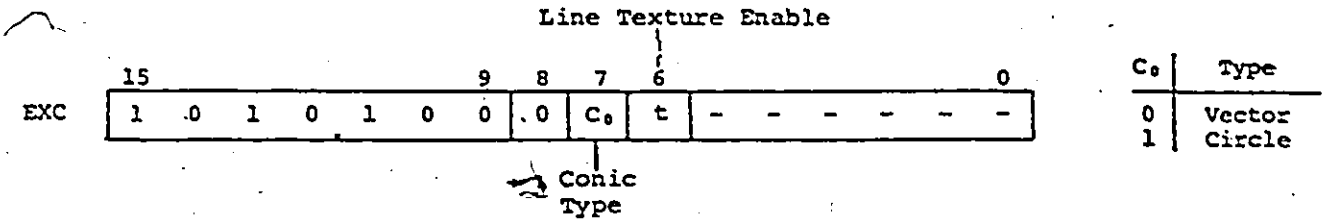
Load Pixel Value - Background *EXT



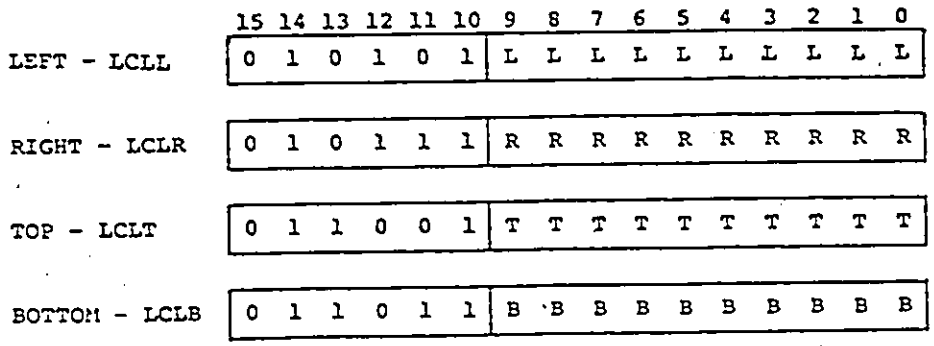
Load Graphic Elements - LGE



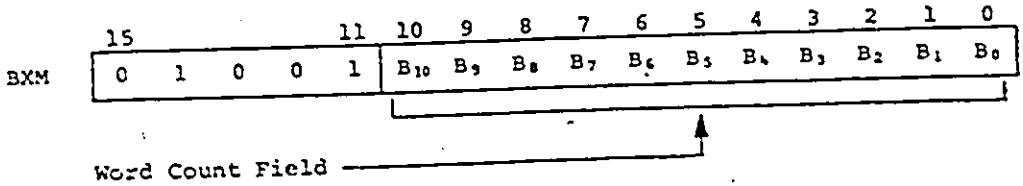
Execute Conic - EXC



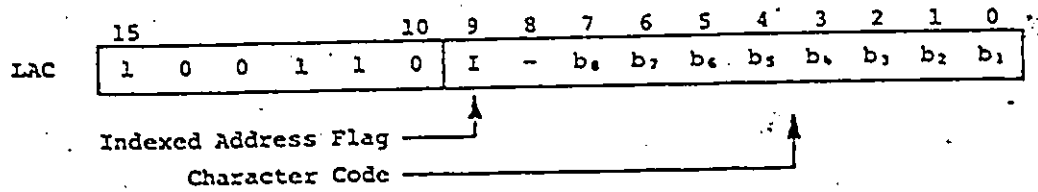
Load Conic Limits - LCLL, LCLR, LCLT, LCLB



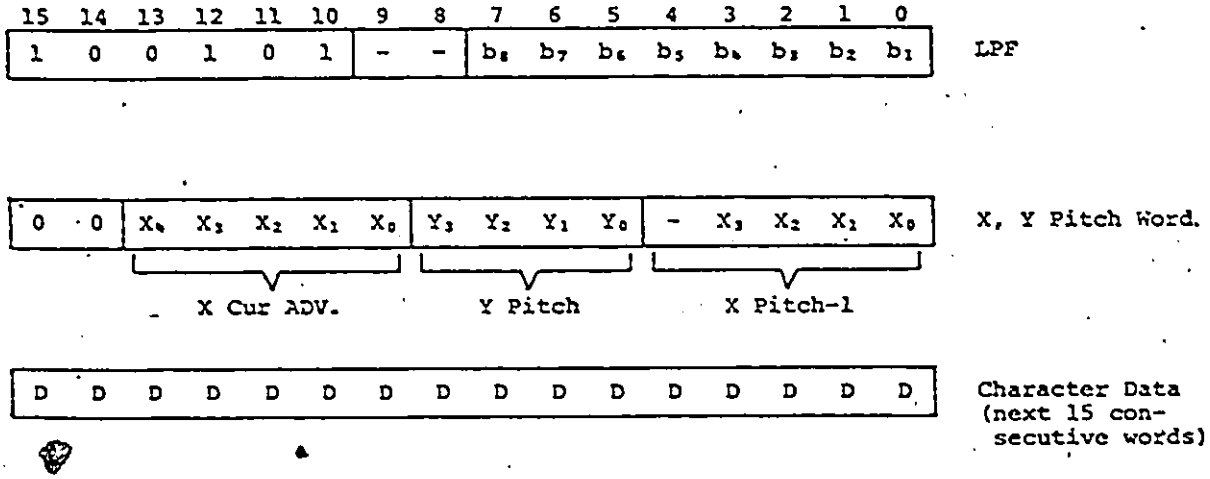
Block Transfer Mode - BXM



Load Alphanumeric Character - LAC



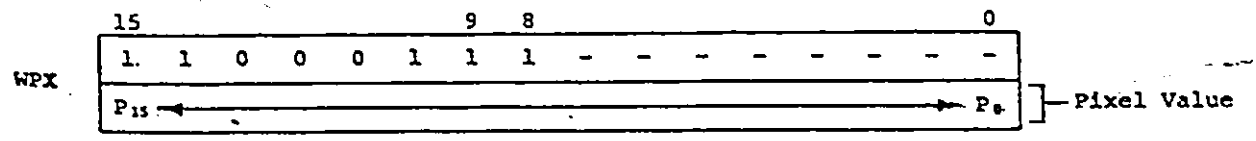
Load Programmable Font - LPF



Write Pixel Values - WPX, WPXF, WPXB

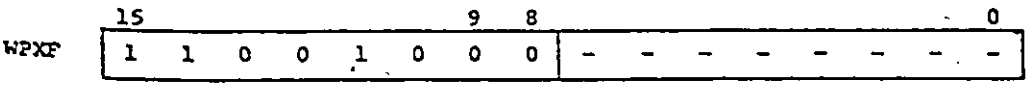
* EXT

Write Pixel



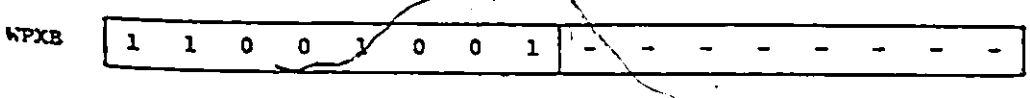
Write Pixel - Foreground

* EXT

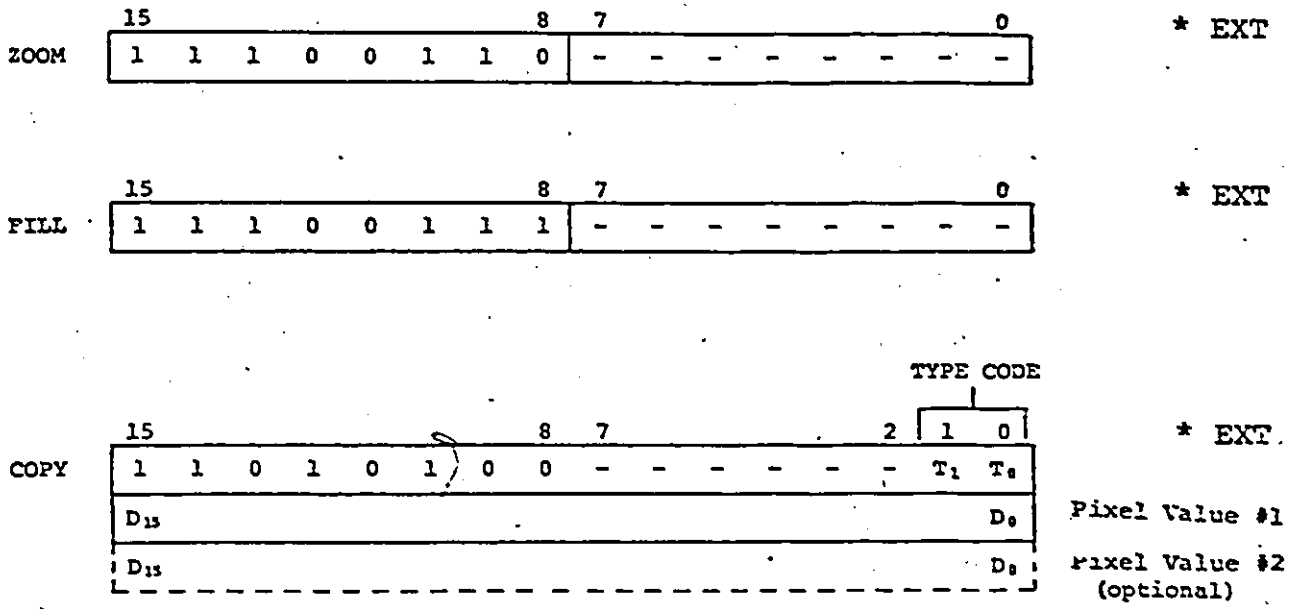


Write Pixel - Background

* EXT



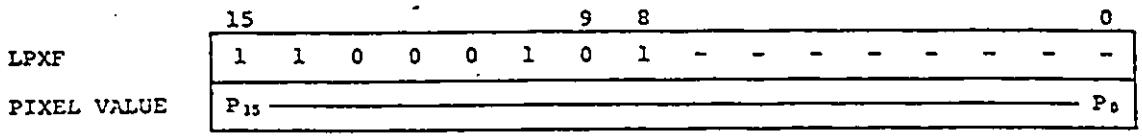
Zoom, Fill, Copy



T_1	T_0	Number of Parameter Words	Will Copy
0	0	1	non-zero after AND
0	1	1	equals
1	0	2	in range
1	1	2	not in range

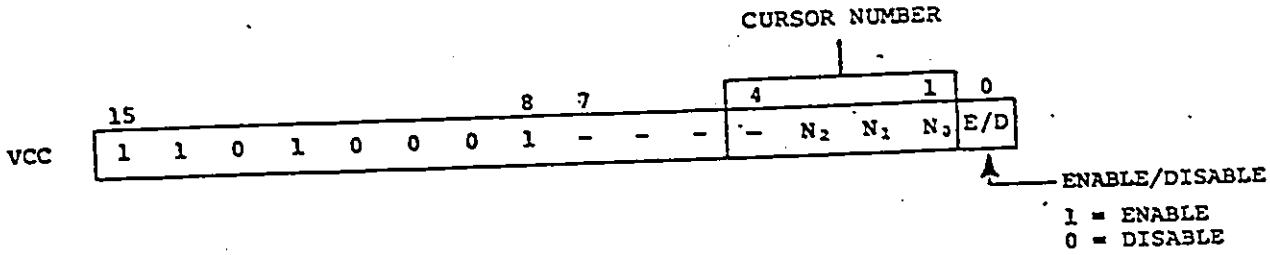
Load Pixel Registers

Load Pixel Value - Foreground *EXT

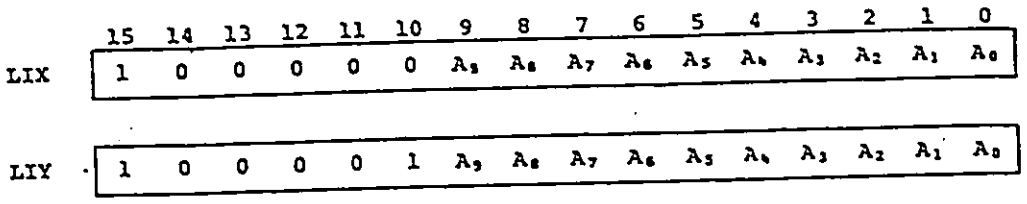


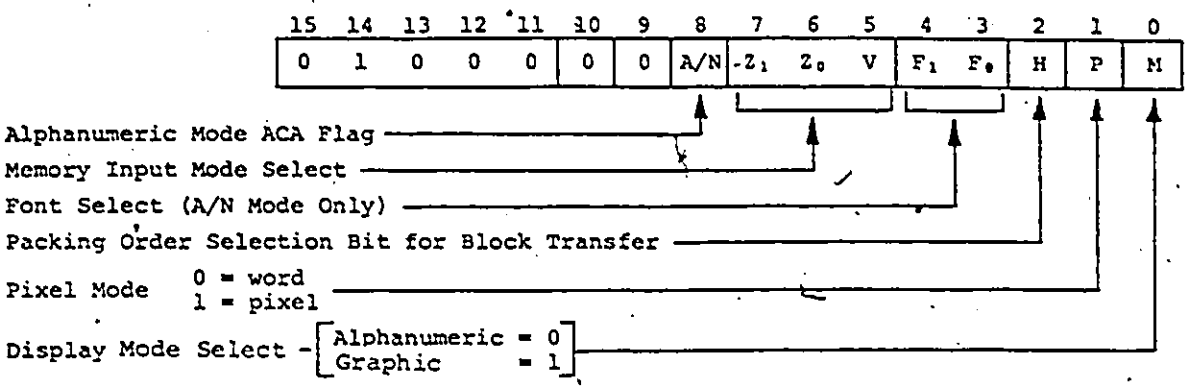
Visible Cursor Control - VCC

*EXT



Load Index Registers - LIX, LIY





Packing Order 0 = low then high
1 = high then low

ACA Flag 0 = Font default
1 = use ACA values

Standard Font Cursor Advance

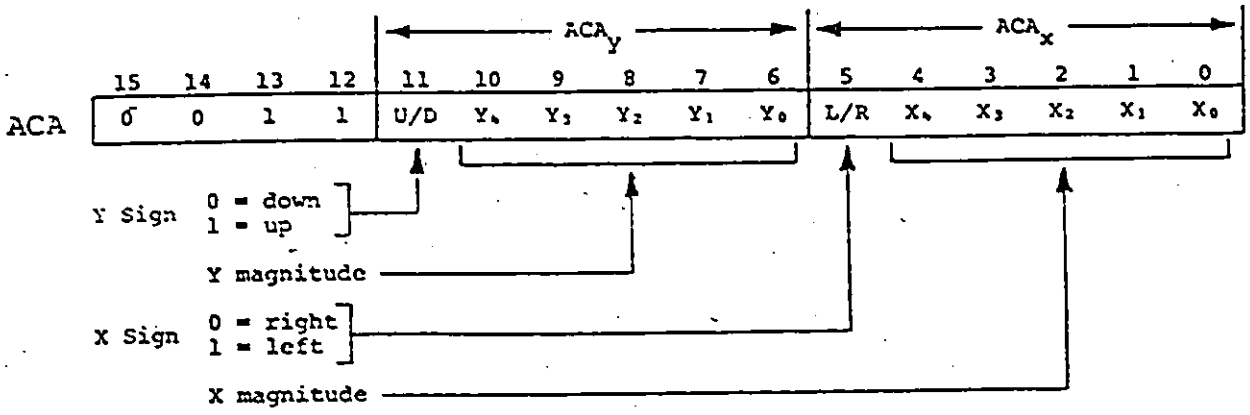
Bit No. 4	3	Font Selected	Cursor Advance in Selected Font (number of picture elements) X	Y
0	0	5 x 7	8	10
0	1	7 x 9	10	14
1	0	10 x 14	16	16
1	1	16 x 15 (Programmable Font)	x-cur.adv.	y-pitch

Bits 3 and 4 have no effect in the graphic mode. The X and Y cursor advances shown are used unless bit 8 of MCW is set.

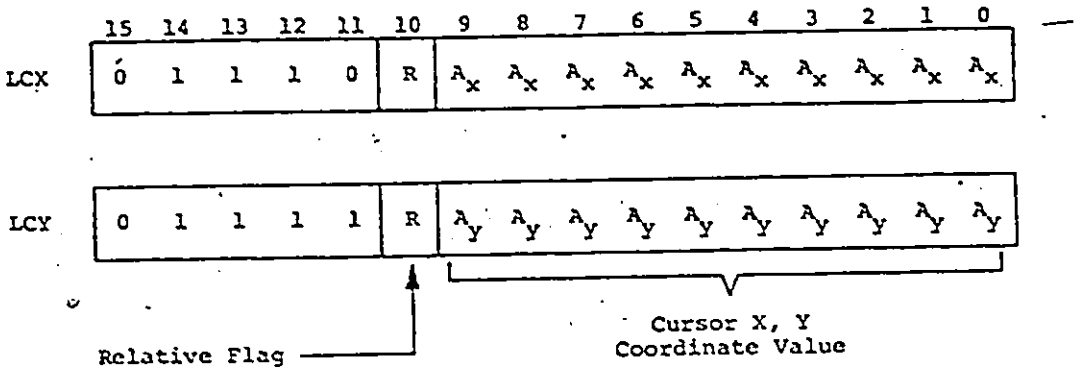
Memory Input Mode Selection

MCW Bit No. 7	6	5 Designation Z_1	Z_0	V	Memory Input Mode
0	0	0			OR Ones
0	1	0			Replace Normal
0	I	1			Replace Reverse
1	0	0			Erase Ones
1	1	X			Not Defined

Adjustable Cursor Advance (ACA)

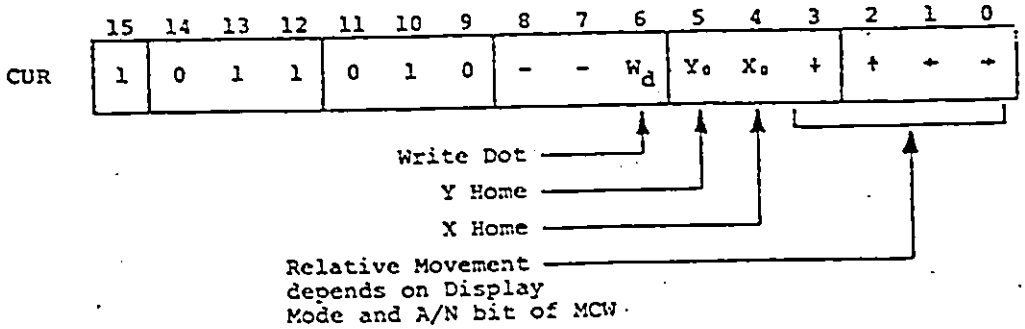


Cursor Positioning - LCX, LCY, CUR



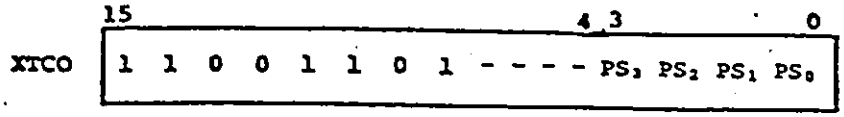
R = 0 Cursor position is specified absolutely

R = 1 Cursor position is specified as an increment relative to the current cursor position

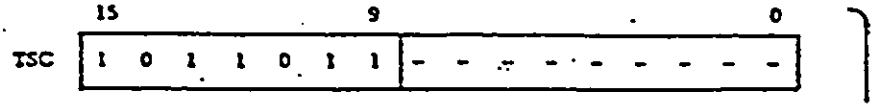


Extended Cursor Position Readback – XTCO

*EXT



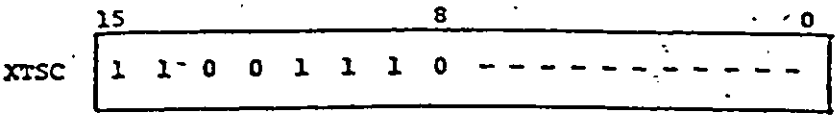
Memory Data Readback – TSC



Extended Memory Data Readback – XTSC

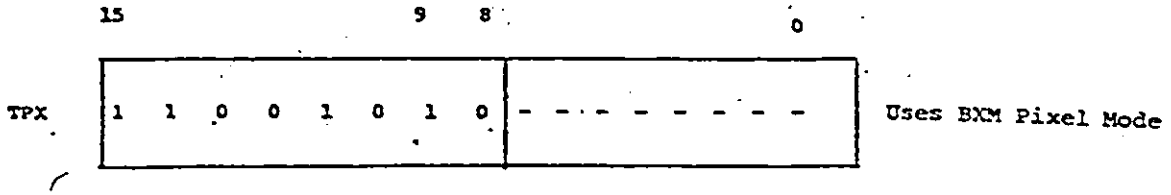
*EXT

Uses BXM Word mode one channel at a time.

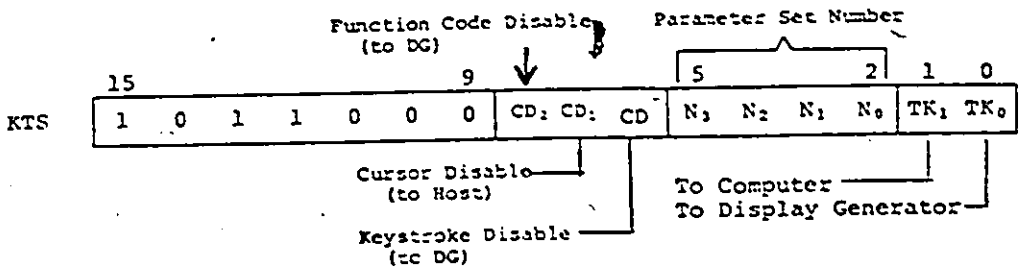


Transmit Picture Elements – TPX

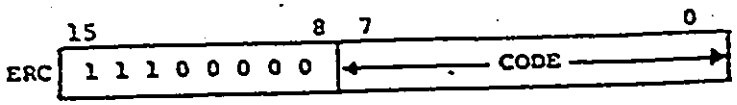
*EXT



Keyboard Transmit Status – KTS



Error Code - ERC



Code	
1	Host/5216 Communications Error
3	Cursor out of limits
8	Illegal (syntax) instruction

Appendix B

PHOTOGRAPHS

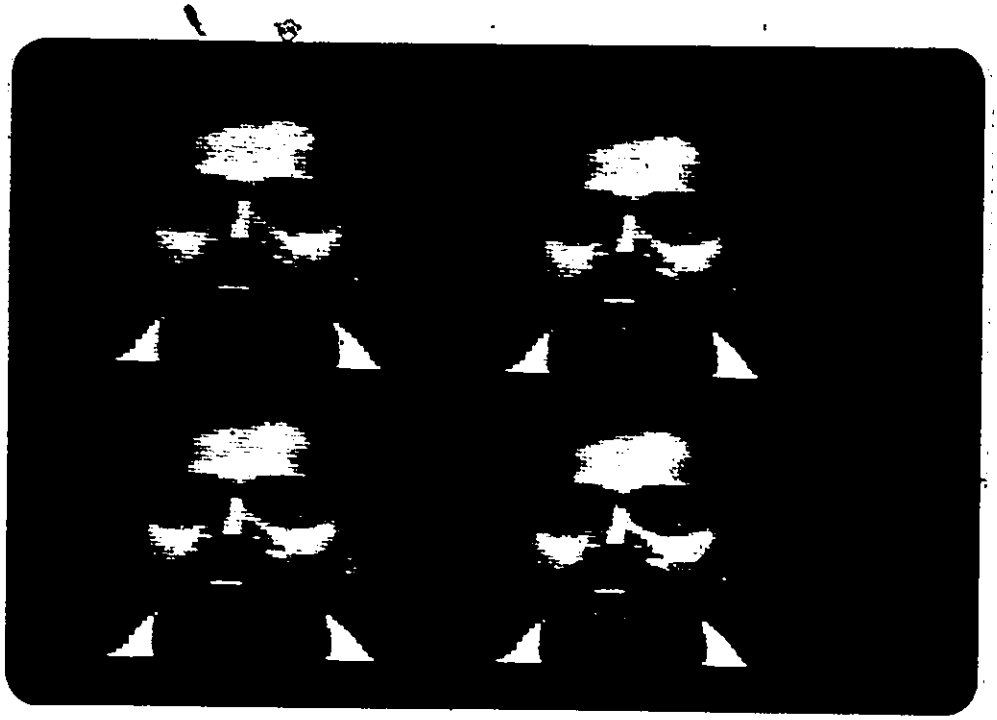


Fig. (A.3.1). : Picture of a face with 16, 32, 64, 256 grey levels clockwise from top left.

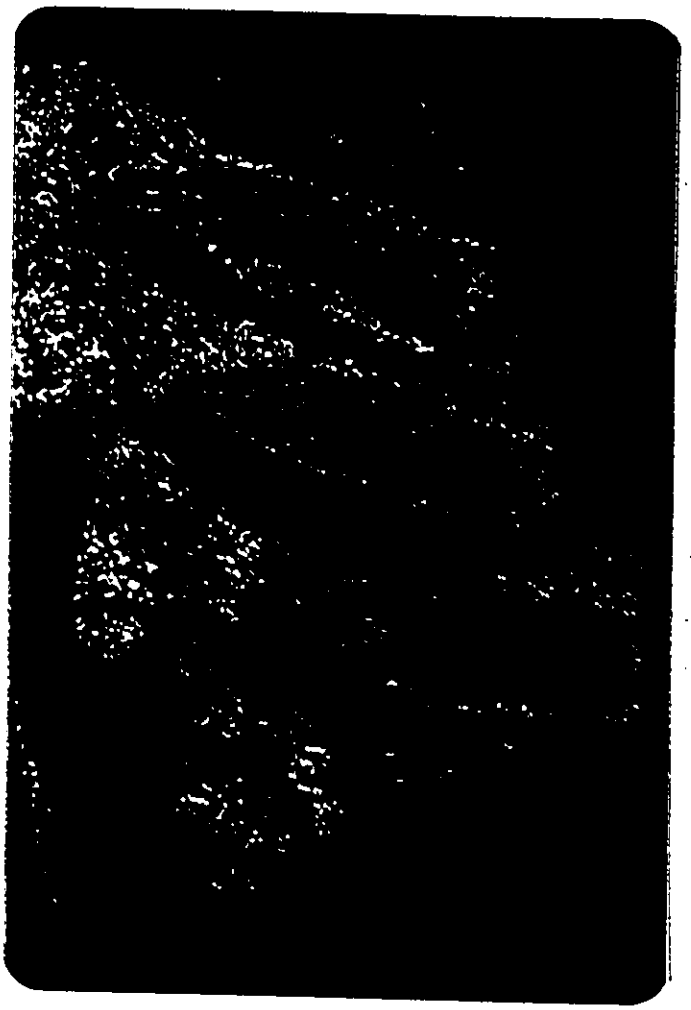


Fig. (A.3.2) : Pseudo Coloured Image

Fig. (A.3.3) : Images in Pseudo colour (Near Real colours)

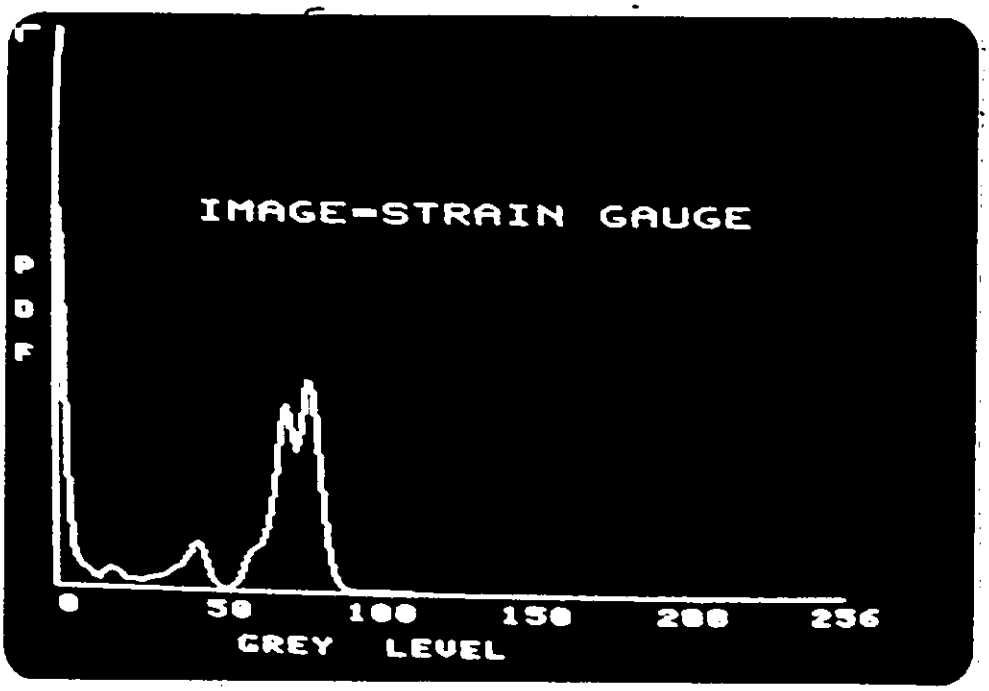


Fig. (A.3.4) : Histogram displayed on The Screen

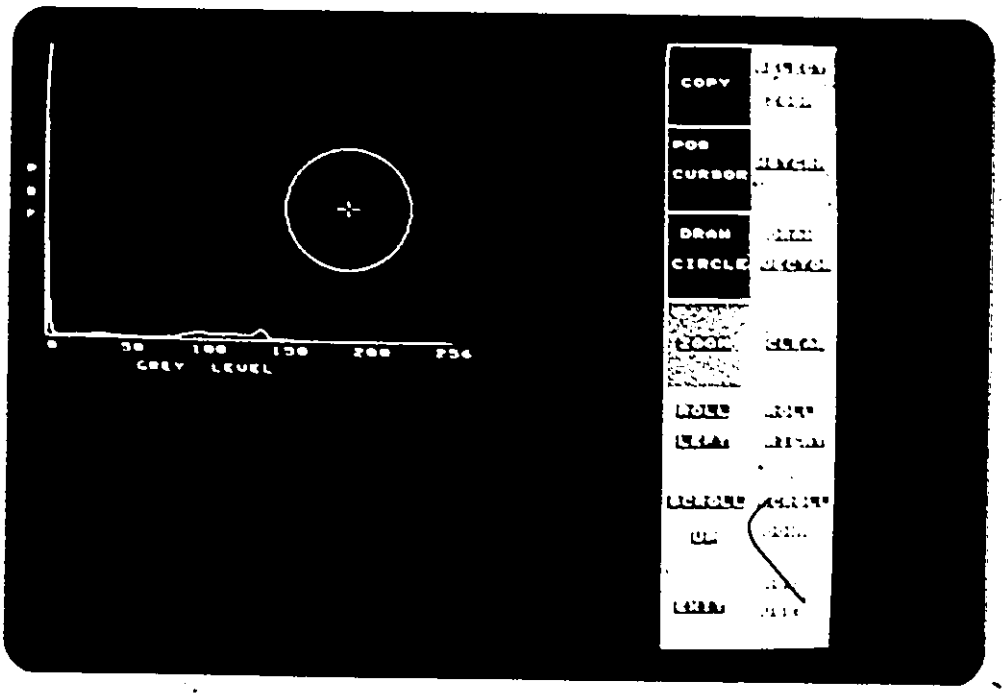
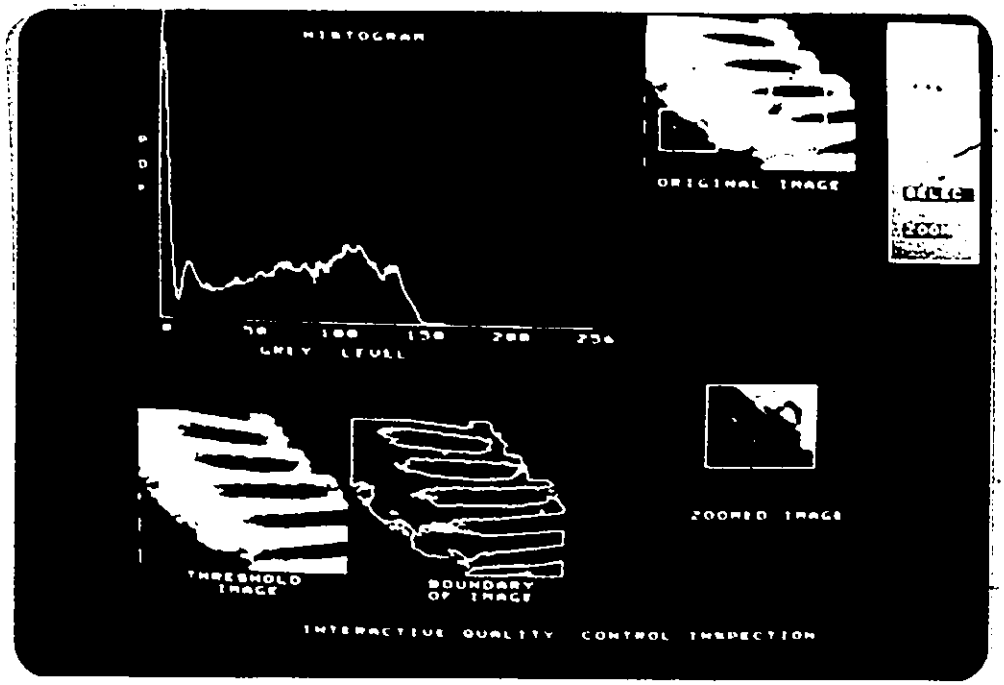
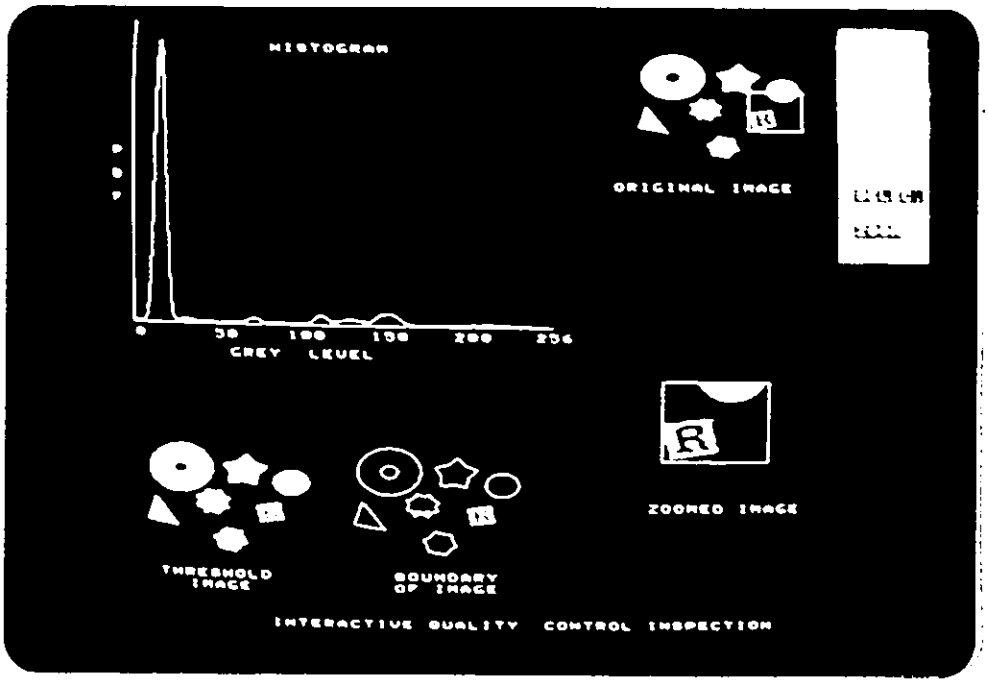


Fig. (A.3.5) : Display Menu with figures drawn through the soft switches displayed on the screen.



(a)



(b)

Fig. (A.3.6) : Display after Program for Interactive Quality Control run on transmission gear and Toys Images. The original Image, Thresholded Image, Boundary of the image, its histogram and a portion selectively zoomed is shown in the pictures.

Fig. (A.3.7) : Images taken with Red , Green, and Blue filters Clockwise from left.

Fig. (A.3.8) : Original Image.

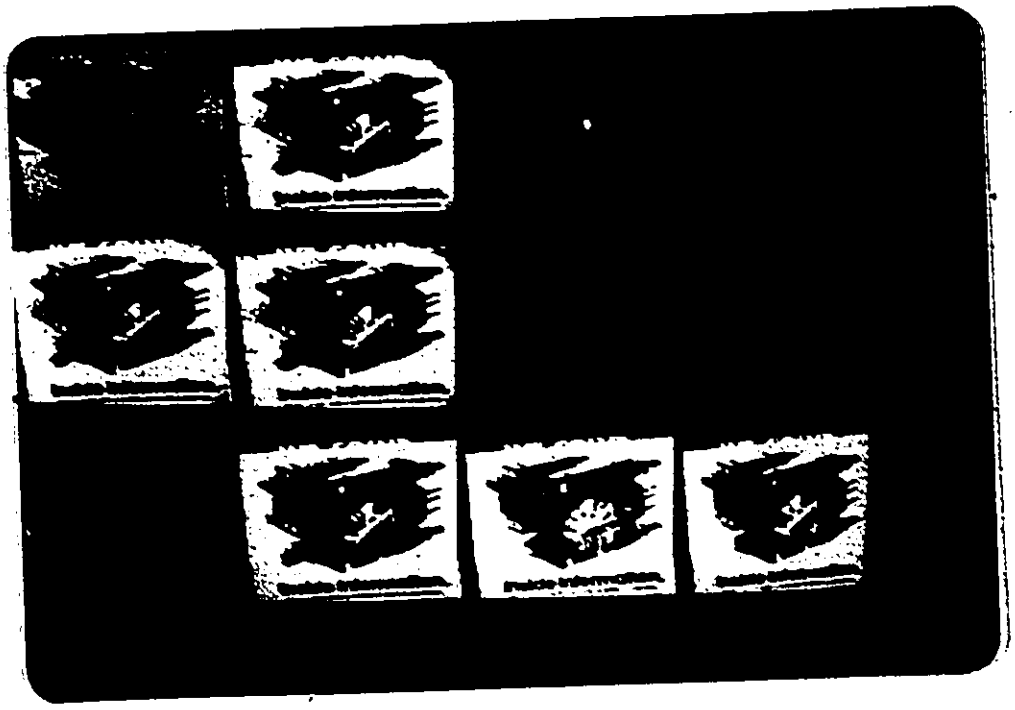


Fig. (A.3.9) : Composite Image.

Appendix C

PROGRAM LISTINGS

```
#JOB TOUCH1 GOYAL SLOF=DUMMY
 #OPTION 1 2 3 4 5 17
 #EXECUTE FORTRAN
 C   ***************************************************************************
 C
 C            PROGRAM FOR THE DISPLAY MENU FOR THE TOUCH SCREEN
 C                 THIS PROGRAM CALLS THE SURROUTINE COORD
 C       TO PASS THE TOUCH SCREEN COORD.  TO THE MAIN PROGRAM
 C
 C   ***************************************************************************
       IMPLICIT INTEGER*2 (A-Z)
       INTEGER*1 IA(128,128), IG, ICHAR(8), IDCHAR(20)
       INTEGER*4 SNSBUF, IERR, TIME, DEVICE, FCB(16)
       REAL PI, AA, BB, HIST(256)
       DIMENSION BUFF1(258), IX(15), IY(15), GL(256), Q(5), N(5)
       DIMENSION BANNER(20)
       EXTERNAL CURPOS, INIT, FIL, EXCCRC, CHAR
 C   ***************************************************************************
 C
 C          INITIALIZE COMMON PARAMETERS
 C
 C   ***************************************************************************
       COMMON/PARSET/TABNUM
       COMMON/MAXBLK/RITMAX, BOTMAX, CHNMAX
       COMMON/CHNBLK/MAJCHN(15), MINCHN(15)
       COMMON/MSKBLK/MAJMSK(15), MINMSK(15)
       COMMON/ACABLK/ACABIT(15)
       COMMON/MONBLK/MONBIT(15)
       COMMON/VNPBLK/RITVIN(15), LFTVIN(15), TOPVIN(15), BOTV)
       COMMON/CONLIM/RITCON(15), LFTCON(15), TOPCON(15), BOTCON(15)
       COMMON/CURBLK/XCUR(15), YCUR(15)
       COMMON/NDXBLK/XINDX(15), YINDX(15)
       COMMON/KTSBLK/KTSBIT(15)
       COMMON/PIXREG/FORPIX(15), BAKPIX(15)
       COMMON/FLGBLK/INDFLG, RELFLG, DOTFLG, ACAFLG(15), GRAFLG
       COMMON/BUFCOM/BUFFLG
       COMMON/ERRCOM/ERLIST(100), ERNUM
       COMMON/OCTBLK/OCTVAL(16)
       COMMON/BUFDAT/BUFARY(500), BUFCNT
       COMMON/FCBS/FCB, SNSBUF
       DATA DEVICE/X'7EA0'/
       DATA HIST/256*0.0/
       TIME=-10
       TABNUM=1
 C   ***************************************************************************
 C
 C          INITIALISE MPCI FOR OPERATION
 C
 C   ***************************************************************************
       CALL IOINIT(DEVICE, IERR, *1)
 C
 C
       CALL SELVIN(1)
     CALL LIMITS(511,-400,511,0)
     CALL ECLR0
```

```
      CALL LIMAX
      CALL CLIMAX
         GOYRL=0
C     ***************************************************************************
C
C                        DRAW THE MENU ON THE SCREEN
C
C     ***************************************************************************
      CALL POSXY(402,0)
      CALL TTT(511,0)
      CALL POSXY(402,0)
      CALL TTT(402,511)
      CALL POSXY(456,0)
      CALL TTT(456,511)
      CALL POSXY(402,73)
      CALL HORVCT(109)
      CALL POSXY(402,146)
      CALL HORVCT(109)
      CALL POSXY(402,219)
      CALL HORVCT(109)
      CALL POSXY(402,292)
      CALL HORVCT(109)
      CALL POSXY(402,365)
      CALL HORVCT(109)
      CALL POSXY(402,438)
      CALL HORVCT(109)
      CALL POSXY(402,511)
      CALL TTT(511,511)
      I=30
      DO 41 J=11,500,73
      CALL POSXY(410,J)
      CALL PIXFOR(I)
      CALL FIL
      I=I+10
41    CONTINUE
      DO 42 J=11,500,73
      CALL POSXY(460,J)
      CALL PIXFOR(I)
      CALL FIL
      I=I+10
42    CONTINUE
      CALL MCNOUT(66)
29       READ(5,*)COL
      IF(COL.EQ.0) GOTO 117
      IF(COL.EQ.1)GOTO 19
      IF(COL.EQ.2) GOTO 21
         READ(5,*)(Q(I),N(I),I=1,5)
         CALL CURPOS(Q(1),N(1))
         DO 43 I=2,5
         CALL TTT(Q(I),N(I))
43       CONTINUE
         XX=Q(1)+5  ;  YY=N(1)+5
         CALL CURPOS(XX,YY)
         CALL PIXFOR(COL)
         CALL FIL
```

```
19      CALL PIXFOR(255)
        READ(5,*)LENGTH,XX,YY
        READ(5,82)(IDCHAR(I),I=1,LENGTH)
82      FORMAT(20A1)
        CALL CURPOS(XX,YY)
        DO 72 I=1,LENGTH
        BANNER(I)=IDCHAR(I)
72      CONTINUE
        CALL RCHAR(LENGTH,BANNER)
        GO TO 29
117     CONTINUE
        IF(GOYAL.EQ.1) GOTO 21
C     ***************************************************************
C
C       GET THE TOUCH COORDINATES AND GO TO THE APPROPRIATE
C                       SECTION OF THE PROGRAM
C
C     ***************************************************************
21    CALL COORD(X,Y)
      CALL LIMAX
      CALL LIMITS(400,0,511,480)
      CALL ECLR0
      CALL LIMAX
      IF (X.LT.400) GOTO 1
      IF((X.LT.456).AND.(Y.LT.73)) GOTO 60
      IF((X.GT.456).AND.(Y.LT.73)) GOTO 110
      IF((X.LT.456).AND.(Y.LT.146)) GOTO 50
      IF((X.GT.456).AND.(Y.LT.146)) GOTO 120
      IF((X.LT.456).AND.(Y.LT.219)) GOTO 40
      IF((X.GT.456).AND.(Y.LT.219)) GOTO 30
      IF((X.LT.456).AND.(Y.LT.292)) GOTO 20
      IF((X.GT.456).AND.(Y.LT.292)) GOTO 10
      IF((X.LT.456).AND.(Y.LT.365)) GOTO 90
      IF((X.GT.456).AND.(Y.LT.365)) GOTO 100
      IF((X.LT.456).AND.(Y.LT.438)) GOTO 70
      IF((X.GT.456).AND.(Y.LT.438)) GOTO 80
      IF((X.LT.456).AND.(Y.LT.511)) GOTO 2
      IF((X.GT.456).AND.(Y.LT.511)) GOTO 200
C     ***************************************************************
C
C                 FILLING A CLOSED AREA WITH COLOUR
C
C     ***************************************************************
200   CALL COORD(X,Y)
      CALL POSXY(X,Y)
      CALL PIXFOR(80)
      CALL FIL
      CALL PIXFOR(255)
      GOTO 21
C     ***************************************************************
C                         CLEARING OF SCREEN
C     ***************************************************************
10    CALL LIMITS(400,0,511,0)
      CALL ECLR0
      CALL LIMAX
```

```
      GOTO 21
C     ************************************************************************
C                              ZOOM THE IMAGE
C     ************************************************************************
20    CALL CLIMS(192,64,192,64)
      CALL POSXY(64,64)
      CALL RCOPY(2,1,255)
      CALL CLIMAX
      CALL LIMITS(256,0,256,0)
      CALL ZOM
      CALL LIMAX
      GOTO 21
C     ************************************************************************
C                              VECTOR DRAWING
C     ************************************************************************
30    CALL COORD(X,Y)
      CALL POSXY(X,Y)
31    CALL COORD(X,Y)
      IF((X.GT.400).AND.(Y.GT.440)) GOTO 21
      CALL TTT(X,Y)
      GOTO 31
C     ************************************************************************
C                              CIRCLE DRAWING
C     ************************************************************************
40    CALL COORD(X,Y)
      CALL CLIMS(400,0,511,0)
      CALL POSXY(X,Y)
48    CALL COORD(A,B)
      IF((A.GT.400).AND.(B.GT.440)) GOTO 49
      Z=IABS(X-A)**2+IABS(Y-B)**2
      AA=Z
      AA=SQRT(AA)
      Z=AA
      CALL RADIUS(Z)
      CALL EXCORC
      GOTO 48
49    CALL CLIMAX
      GOTO 21
C     ************************************************************************
C                              POSITIONING OF CURSOR
C     ************************************************************************
50    CALL COORD(X,Y)
      CALL POSXY(X,Y)
      GOTO 21
C     ************************************************************************
C                        COPY IMAGE TO ANOTHER SECTION
C     ************************************************************************
60    CALL COORD(X,Y)
      CALL POSXY(X,Y)
      CALL COORD(X,Y)
      CALL COORD(R,B)
      IF(X.LT.R) GOTO 61
      A=R
      R=X
      X=A
```

```
      A=Y
      Y=B
      B=A
61    CALL CLIMS(R,X,B,Y)
      CALL RCOPY(2,1,255)
      CALL CLIMAX
      GOTO 21
C     ************************************************************************
C                          SCROLLING UP THE IMAGE
C     ************************************************************************
70    CALL LIMITS(400,0,511,0)
      CALL NCSRUP(5)
      CALL LIMAX
      GOTO 21
C     ************************************************************************
C                          SCROLLING DOWN THE IMAGE
C     ************************************************************************
80    CALL LIMITS(400,0,511,0)
      CALL NSCDWN(5)
      CALL LIMAX
      GOTO 21
C     ************************************************************************
C                          ROLL IMAGE LEFT
C     ************************************************************************
90    CALL LIMITS(400,0,511,0)
      CALL NROLLF(5)
      CALL LIMAX
      GOTO 21
C     ************************************************************************
C                          ROLL IMAGE RIGHT
C     ************************************************************************
100   CALL LIMITS(400,0,511,0)
      CALL NROLRT(5)
      CALL LIMAX
      GOTO 21
C     ************************************************************************
C                          SELECTIVE ZOOMING OF THE IMAGE
C     ************************************************************************
110   CALL COORD(X,Y)
      IF((X.GT.400).AND.(Y.GT.440)) GOTO 112
      A=X-17
      B=Y-17
      CALL POSXY(A,B)
      CALL PIXFOR(255)
      CALL SQUARE(34)
      CALL COORD(C,D)
      IF((C.GT.400).AND.(D.GT.440)) GOTO 112
      E=C-33
      F=D-33
      SE=E-1
      SF=F-1
      CALL POSXY(SE,SF)
      CALL SQUARE(68)
      CALL POSXY(X,Y)
      G=C-16
```

```
      K=D-16
      I=C+16
      J=D+16
      CALL CLIMS(I,G,J,H)
      CALL RCOPY(2,1,255)
      CALL CLIMAX
      K=C+33
      L=D+33
      CALL LIMITS(K,E,L,F)
      CALL ZOM
      CALL LIMAX
      CALL CLIMAX
      GOTO 110
  112 CALL LIMAX
      CALL CLIMAX
      GOTO 21
C     ***************************************************************
C                    HISTOGRAM PLOT OF IMAGE SIZE
C                              OF 128 * 128
C     ***************************************************************
  120 NX=128; NY=128
      GOYAL=0
      OPEN(UNIT=2,BLOCKED=.FALSE.,FORM='UNFORMATTED')
      READ(2)((IA(I,J),J=1,NY),I=1,NX)
      CLOSE(UNIT=2)
      DO 17 I=1,NX
      DO 17 J=1,NY
      IG=IA(I,J)+1
      HIST(IG)=HIST(IG)+1
   17 CONTINUE
      AA=0; BB=100000000
      DO 12 I=1,256
      GL(I)=I-1
      AA=AMAX1(HIST(I),AA)
   12 BB=AMIN1(HIST(I),BB)
      DIFF=AA-BB
      DO 23 I=1,256
      HIST(I)=HIST(I)*256/DIFF
   23 CONTINUE
C     ***************************************************************
C       SMOOTHING THE HISTOGRAM
C     ***************************************************************
        NP=-1
        NN=+2
        DO 999 I=2,255
        I1=I+NP; I2=I+NN
        HIST(I)=(HIST(I1)+HIST(I)+HIST(I2))/3
  999   CONTINUE
      CALL POSXY(15,0)
      CALL TTT(15,256)
      CALL TTT(271,256)
      CALL POSXY(15,256)
      DO 22 I=1,256
      IX=GL(I)+15
      IY=256-HIST(I)
```

```
      CALL PIXFOR(255)
      CALL TTT(IX,IY)
22    CONTINUE
      GOYAL=1
      GOTO 29
2     CALL CLIMAX
      CALL LIMAX
         STOP
1        TYPE*,'ERROR ENCOUNTERED'
      TYPE*,'"07'
      CALL NFLRG
      INR=0
1001  INR=INR+30
      CALL POSXY(INR,480)
      CALL CHAR(69)
      INR=INR+17
      CALL POSXY(INR,480)
      CALL CHAR(82)
      INR=INR+17
      CALL POSXY(INR,480)
      CALL CHAR(82)
      INR=INR+17
      CALL POSXY(INR,480)
      CALL CHAR(79)
      INR=INR+17
      CALL POSXY(INR,480)
      CALL CHAR(82)
      IF(INR.LE.280) GOTO 1001
      CALL NFSML
      GOTO 21
      END
      INCLUDE COORD
$AS LIB TO @SYSTEM(SYSTEM)AYDLIB BLOCKED=N
$AS DIR TO @SYSTEM(SYSTEM)AYDDIR BLOCKED=N
$EXECUTE CATALOG
AS 1=CA7EC7
AS 2 TO @SYSTEM(IMAGE)PST2
A1 S=TOUCHDAT
A4 6=UT
BUILD TOUCHSV,NOM,P
$EOJ
$$
```

```
C     ***********************************************************
C                          SUBROUTINE COORD
C
C
C              THIS ROUTINE RETURNS THE TOUCH POSITION TO
C              THE MAIN PROGRAM.  THE VARIABLES ARE X AND Y
C     ***********************************************************
      SUBROUTINE COORD(X,Y)
      IMPLICIT INTEGER*2 (A-Z)
      INTEGER*1 ICHAR(8)
      READ (1,3) (ICHAR(I),I=1,8)
3     FORMAT(8A1)
      DO 4 I=2,7
4     ICHAR(I)=ICHAR(I)-X'30'
      X=ICHAR(2)*100+ICHAR(3)*10+ICHAR(4)
      Y=ICHAR(5)*100+ICHAR(6)*10+ICHAR(7)
      X=X*2
      Y=Y*2
      RETURN
      END
```

REFERENCES

1. Digital Image Processing, pp 174 - 176, Gonzalvez, Rafel C. and Wintz, Paul, Addison Wesley Publishing Company, Inc., 1977

2. 5216 Standard Firmware Fortran Support Package Users Manual, Aydin Controls, Document Number 150-6045-114, May 1980

3. 5216 Standard Firmware Users Manual, Aydin Controls, Document Number 150-6045-004, Revised Nov. 1982

4. Multipurpose Custom Interface Users Guide, Gould SEL Computer Systems Division, Publication Number 321-000840-000, June 1982

5. Technical Manual, Input/Output Processor (IOP) Model 8000 and I/O Expansion Chasis Model 8910, Gould SEL Computer Systems Div., Publication No. 303-328000-001, March 1981

6. MPX - 32 (Release 2.1) Reference Manual, Volume I and II, ibid, Publication No. 323-001011-301, August 1982

7. Reference Manual, FORTRAN - 77+, Release 3.0, Change I, ibid, Publication No. 323-001470-001, November 1981

8. Project Research Applicable in Industry NSERC File # 612 - 14/79, 1983.

9. The Role of Human Visual Models in Image Processing, Douglas J. Granrath, Proc IEEE, Vol 69, No. 5, May 1981

10. Image Processing by Digital Computer, Andrew H.C.,Tescher A.G.,Kruger R.P.,IEEE Spectrum, Vol. 9, No. 7, PP 20-32,1972

11. A Pseudo Colour Image Printing System, A Thesis presented to The University of Windsor Through The Department of Electrical Engineering, 1984.

12. Digital Image Processing, William K. Pratt, A Willey - Interscience publication , 1978

13. Digital Image Processing, Kenneth R. Castleman, Prentice Hall Signal Processing Series

14. Design and Implementation of a High Speed Video Digitizer A Thesis presented to the University of Windsor through the Department of Electrical Engineering, 1984

15. Algorithms for Graphics and Image Procesing, Pavlidis, Theo, Computer Science Press, 1982

16. Color Coding and Visual Separability in Information Displays, Smith, S.L., J. Appl. Psychology, Vol 47, PP 358-364

17. Touch screen , TSD Display Products, Inc.

VITA AUCTORIS

PRADEEP KUMAR GOYAL

1960 Born on 10th August in Ambala, India

1976 Completed All India Higher Secondry at Central School Delhi Cantt, India

1981 Graduated from Birla Institute of Technolcgy with B.Sc (Electrical Engg.) Degree

1984 Candidate for Degree of Master of Applied Science in Department of Electrical Engineering at Oriversity of Windsor, Windsor, Ontario, Canada